Face Recognition Using Feature Fusion and Deep Learning

by

Xin Wei

A THESIS SUBMITTED TO ULSTER UNIVERSITY
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY
IN THE SCHOOL OF COMPUTING
FACULTY OF COMPUTING, ENGINEERING
AND THE BUILT ENVIRONMENT
May 2020

DECLARATION

I confirm that the word count of this thesis is less than 100,000 excluding the title page, contents acknowledgements, summary or abstract, abbreviations, footnotes, diagrams, maps, illustrations, tables, appendices, and references or bibliography.

I hereby declare that with effect from the date on which the thesis is deposited in Research Student Administration of Ulster University, I permit:

(i). The Librarian of the University to allow the thesis to be copied in whole or in part without reference to me on the understanding that such authority applies to the provision of single copies made for study purposes or for inclusion within the stock of another library.

(ii). The thesis to be made available through the Ulster Institutional Repository and/or EThOS under the terms of the Ulster eTheses Deposit Agreement which I have signed.

IT IS A CONDITION OF USE OF THIS THESIS THAT ANYONE WHO CONSULTS IT MUST RECOGNISE THAT THE COPYRIGHT RESTS WITH THE AUTHOR AND THAT NO QUOTATION FROM THE THESIS AND NO INFORMATION DERIVED FROM IT MAY BE PUBLISHED UNLESS THE SOURCE IS PROPERLY ACKNOWLEDGED.

Student: Xin Wei

Data: 20/04/2020

ACKNOWLEDGEMENTS

First and foremost, I'd like to thank my first supervisor Prof. Hui Wang. This long journey has only been possible due to his constant support and encouragement. He encouraged me to unleash my potential, helped me to discover research opportunities, and overcame countless challenges with me. He was always the source of thoughtful advice, full confidence and contagious enthusiasm.

I'd like to thank my second supervisor Prof. Bryan Scotney for his patience, support and guidance during my research studies. He brought me a rigorous academic attitude. He taught me how to solve all kinds of difficulties with a calm mind and taught me quite a lot in academic writing.

Many thanks to all my friends who morally supported me throughout my life and study. Miss the days of exercising, gathering and playing with you.

Last but not least, special thanks to my wife and parents. They are always there to support me in my study. They keep me accompany and bring me joy and happiness. They gave me spiritual encouragement when I was confused and made me keep going when I was tired.

PUBLICATIONS

Journals:

[1]. **Xin Wei**, Hui Wang, Bryan Scotney and Huan Wan. Minimum Margin Loss for Deep Face Recognition. Pattern Recognition. 2019: 107012.

[2]. **Xin Wei**, Hui Wang, Bryan Scotney and Huan Wan. Selective multi-descriptor fusion for face identification. International Journal of Machine Learning and Cybernetics. 2019: 1-13.

[3]. Huan Wan, Hui Wang, Gongde Guo and **Xin Wei**. Separability-oriented subclass discriminant analysis. IEEE transactions on pattern analysis and machine intelligence. 2017 Feb 22; 40(2): 409-22.

Conferences:

[1]. **Xin Wei**, Hui Wang, Bryan Scotney and Huan Wan. Precise Adjacent Margin Loss for Deep Face Recognition. IEEE International Conference on Image Processing (ICIP) 2019, Sep 22 (pp. 3457-3461).

[2]. **Xin Wei**, Hui Wang, Bryan Scotney and Huan Wan. GicoFace: Global Information-based Cosine Optimal Loss for Deep Face Recognition. IEEE International Conference on Image Processing (ICIP) 2019, Sep 22 (pp. 3641-3645).

[3]. **Xin Wei**, Hui Wang, Bryan Scotney and Huan Wan. A Minkowski Distance-based Generalisation Method for Improving Centre Loss for Deep Face Recognition **(Best**

Student Paper Award). Irish Machine Vision and Image Processing Conference (IMVIP) 2018, Aug (pp. 154-161). IPRCS.

[4]. **Xin Wei**, Hui Wang, Huan Wan and Bryan Scotney. Self-adaptive Feature Fusion Method for Improving LBP for Face Identification. International Conference on Computer Vision Systems (ICVS) 2017, Jul 10 (pp. 373-383). Springer, Cham.

[5]. **Xin Wei**, Hui Wang, Huan Wan and Bryan Scotney. Multiscale Feature Fusion for Face Identification. IEEE International Conference on Cybernetics (CYBCONF) 2017, Jun 21 (pp. 1-6).

TABLE OF CONTENTS

DECLARATION ii

ACKNOWLEDGEMENTS iii

PUBLICATIONS iv

LIST OF FIGURES ix

LIST OF TABLES xi

LIST OF ABBREVIATIONS xiii

ABSTRACT xvi

CHAPTER

I. Introduction 1

1.1 Overview of Face Recognition 1
1.2 Research Motivation and Contributions 5
1.3 Outline of the Thesis 9

II. Face Image Representation 11

2.1 Introduction 11
2.2 Hand-crafted Face Representation 12
2.2.1 Local Binary Patterns and Its Variants 12
2.2.2 Histogram of Oriented Gradients 13
2.2.3 Hidden Markov Model 14
2.3 Shallow Learning-based Representation 15
2.3.1 Subspace Learning 16
2.3.2 Sparse Coding 17
2.3.3 BoW Model 18
2.3.4 Shallow Neural Networks 19
2.4 Deep Learning-based Representation 20

2.4.1 Deep Neural Networks . . . 20
2.4.2 Network Architectures . . . 21
2.4.3 Loss Functions . . . 24
2.5 Summary . . . 26

III. Selective Multi-descriptor Fusion for Face Identification . . . 27

3.1 Introduction . . . 27
3.2 Related Work . . . 28
3.3 Selective Multi-descriptor Fusion . . . 31
3.3.1 Initial Feature Extraction . . . 31
3.3.2 Descriptor Selection . . . 34
3.4 Experiments . . . 40
3.4.1 Results on the CAS-PEAL-R1 Dataset . . . 40
3.4.2 Results on the LFW Dataset . . . 44
3.4.3 Stability and Runtime Evaluation . . . 46
3.5 Summary . . . 48

IV. Euclidean Distance-based Losses in Deep Face Recognition . . . 49

4.1 Motivation . . . 49
4.2 A Minkowski Distance-based Method for Deep Face Recognition . . . 50
4.2.1 Softmax Loss and Centre Loss . . . 51
4.2.2 The Proposed Minkowski Distance-based Centre Loss . . . 52
4.2.3 Experiments . . . 55
4.3 Minimum Margin Loss for Deep Face Recognition . . . 57
4.3.1 Marginal Loss and Range Loss . . . 58
4.3.2 The Proposed Minimum Margin Loss . . . 60
4.3.3 Discussion . . . 61
4.3.4 Experiments . . . 63
4.4 Summary . . . 73

V. Cosine Similarity-based Losses in Deep Face Recognition . . . 74

5.1 Motivation . . . 74
5.2 Precise Adjacent Margin Loss for Deep Face Recognition . . . 75
5.2.1 Related Work . . . 75
5.2.2 The Proposed PAM Loss . . . 77
5.2.3 Experiments . . . 81
5.3 GICO Loss for Deep Face Recognition . . . 84
5.3.1 The Proposed GICO Loss . . . 84
5.3.2 GICO Lite A . . . 87
5.3.3 GICO Lite B . . . 89
5.3.4 GICO Std and Discussion . . . 90
5.3.5 Experiments . . . 91

5.4 Summary 95

VI. A Complete Face Search Framework for Image and Video Retrieval . . 96

6.1 Introduction 96
6.2 Motivation 97
6.3 The Framework 97
6.4 A Demonstrator 102
6.5 Functional Evaluation 106
6.5.1 High Retrieval Accuracy 106
6.5.2 Low Requirement on Uploaded Image 110
6.5.3 Support Face Search in Videos 112
6.6 Summary 112

VII. Conclusions and Future Work 114

7.1 Conclusions 114
7.2 Future Work 116

REFERENCES 118

LIST OF FIGURES

Figure

1.1 The basic framework of a face recognition system. 3

2.1 Overview of HOG. 14

2.2 Five-state HMM for face recognition. 15

2.3 The basic structure of SNN-based representation. 19

2.4 A simple deep neural network. 20

3.1 The pipeline of the proposed SMDF. 32

3.2 Landmarks located by SDM and the landmark-based local features. 33

3.3 Illustration of the dual-cross encoder. 34

3.4 Examples from the CAS-PEAL-R1 dataset. 40

3.5 Examples from the LFW dataset. 44

3.6 Identification accuracy VS number of descriptors 47

4.1 Examples from the LFW and YTF dataset. 56

4.2 The effectiveness of the proposed MML. 62

4.3 Total loss of two groups of models. 66

4.4 CMC and ROC curves of different methods on MegaFace Set 1. 67

4.5 Examples from the LFW and the YTF dataset. 69

4.6 Examples of the negative pairs in LFW and SLLFW. 71

5.1 Geometrical interpretation of the Softmax Loss and the real margin. . . . 77

5.2 The CMC and ROC curves of different methods on MegaFace. 83

5.3 An overview of the proposed GicoFace framework. 86

5.4 The CMC and ROC curves of different methods on MegaFace. 92

6.1 Flowchart of the proposed face search framework. 98

6.2 Cover page of the proposed face search framework. 99

6.3 Example of face search on images. 100

6.4 Example of face search in videos. 101

6.5 The web page for adding more videos. 102

6.6 Example of the Youtube videos in the gallery. 104

6.7 Example of the preprocessed faces. 104

6.8 Examples of image search results of different frameworks. 107

6.9 Results of different indexing methods on CIFAR dataset. 109

6.10 Search results of different face search framework on the same uploaded image. 110

6.11 Results of different face detection methods on three subsets of the WIDER FACE dataset. 111

6.12 Five examples of face search results in videos. 112

LIST OF TABLES

Table

3.1 Comparison with SRC-based methods on CAS-PEAL-R1. 42

3.2 Comparison with other descriptors on the CAS-PEAL-R1 dataset. 43

3.3 Comparison with state-of-the-art methods on six subsets of the CAS-PEAL-R1 dataset. 43

3.4 Comparison with state-of-the-art methods on LFW. 45

3.5 Runtime evaluation on LFW dataset. 48

4.1 Statistics for recent public available large-scale face datasets. 50

4.2 Verification performance of state-of-the-art methods on LFW and YTF. . . 57

4.3 The identification rates and the verification rates of different methods. . . 67

4.4 Verification rates of state-of-the-art methods on LFW and YTF. 69

4.5 Verification performance of different methods on SLLFW. 71

4.6 Results with 1:1 verification protocol on IJB-B and IJB-C. 72

5.1 Parameter settings about the network and the testing. 81

5.2 Verification accuracy of the methods on LFW and YTF. 83

5.3 Properties of different loss functions in deep face recognition. 85

5.4 Verification performance of different methods on LFW and YTF. 93

5.5 Verification performance of different methods on SLLFW. 94

6.1 Verification rates of state-of-the-art models on LFW and YTF datasets. . . 108

LIST OF ABBREVIATIONS

(Sort in alphabetical order)

AAM	Active Appearance Model
ACF	Aggregate Channel Features
AM-Softmax Loss	Additive Margin Softmax Loss
ASM	Active Shape Model
A-Softmax Loss	Angular Softmax Loss
Auto-ML	Automatic Machine Learning
BNC	Block Number of Each Column
BNR	Block Number of Each Row
BoW	Bag-of-Words
BP	Back Propagation
CCBLD	Chain Code-based Local Descriptor
CFD	Current Feature Dimension
CMC	Cumulative Match Characteristics
CNNs	Convolutional Neural Networks
DA	Discriminant Ability
DAMS	Discriminant Ability-based Multi-descriptor Selection
DBN	Deep Belief Network
DCP	Dual-Cross Patterns
DCT	Discrete Cosine Transforms
DNNs	Deep Neural Networks
ESRC	Extended Sparse Representation-based Classifier
FAR	False Accept Rate
FC layer	fully-connected layer
FDER	Feature Descriptor using Entropy Rate
FRR	False Reject Rate
GICO Loss	Global Information-based Cosine Optimal Loss
GWT	Gabor Wavelet Transforms
HMM	Hidden Markov Model
HOG	Histogram of Oriented Gradient
HR	high-resolution
ICA	Independent Component Analysis
ITQ	Iterative Quantization

I2CDDE	Discriminative Embedding Method based on the Image-to-Class Distance
KDA	Kernel Discriminant Analysis
KED	Kernel Extended Dictionary
kNN	k-Nearest Neighbour
KPCA	Kernel Principal Component Analysis
LBP	Local Binary Patterns
LDA	Linear Discriminant Analysis
LFW	Labeled Faces in the Wild
LGXP	Local Gabor XOR Patterns
LPQ	Local Phase Quantisation
LR	low-resolution
LRFR	Low-resolution Face Recognition
LSH	Locality Sensitive Hashing
L-Softmax Loss	Large-Margin Softmax Loss
MC Loss	Minkowski Distance-based Centre Loss
MCCNN	Multiscale Cascade CNN
MDF	Multi-descriptor Fusion
MML	Minimum Margin Loss
MsTLBP	Multiscale tLBP
MsDLBP	Multiscale dLBP
MTCNN	Multi-Task Cascaded Convolutional Neural Network
NAS	Neural Architecture Search
NN	Neural Network
OSPP	One Sample per Person
PAM Loss	Precise Adjacent Margin Loss
PCA	Principal Component Analysis
PCNN	Principal Component Neural Network
PIFR	Pose-invariant Face Recognition
QLRBP	Quaternionic Local Ranking Binary Pattern
RBFN	Radial Basis Function Network
RIP	Restricted Isometry Property
ROC	Receiver Operating Characteristic
RRC	Regularised Robust Coding
SDM	Supervised Descent Method
SH	Spectral Hashing
SIFT	Scale-invariant Feature Transform
SLF-RKR	Robust Kernel Representation with Statistical Local Features
SLLFW	Similar-looking LFW
SMDF	Selective Multi-descriptor Fusion
SNN	Shallow Neural network
SRC	Sparse Representation Classifier
SSEC	Structured Sparse Error Coding
SSRC	Superposed Sparse Representation Classifier
SVM	Support Vector Machine

TSCNN Two-stage CNN
YTF YouTube Faces

ABSTRACT

Face recognition is a research hotspot in the fields of computer vision and deep learning. Over the past decade, impressive progress has been made, but there are still challenges in face recognition, as the facial portion of an image can easily be affected by various factors like occlusion, age, illumination and pose. A typical face recognition system includes face detection, face image representation and final classification, where the key process is face image representation, namely, face image-based feature extraction. This thesis focuses on learning highly discriminative facial features with traditional machine learning techniques and deep learning techniques to improve face recognition performance.

With traditional machine learning techniques, we firstly propose a feature fusion method called Multi-descriptor Fusion (MDF) in Chapter III to generate hyper-high dimensional descriptor features and highly discriminative features. The performance of MDF is competitive even compared with some deep learning-based methods. In a deep neural network, a loss function plays an extremely important role, which supervises the whole training process and offers feedback information to optimise the parameters of the neural network. Better loss functions can lead to more effective features. Therefore, we propose four loss functions in deep learning-based face recognition, where Minkowski Distance-based Centre Loss (MC Loss) and Minimum Margin Loss (MML) are presented in Chapter IV, and Precise Adjacent Margin Loss (PAM Loss) and Global Information-based Cosine Optimal Loss (GICO Loss) are presented in Chapter V. MC Loss extends the Centre Loss from the Euclidean distance to the Minkowski distance. MML enhances the discriminative ability of features by setting a minimum distance between all pairs of the class centres. PAM Loss penalises the margin and gives 'margin' a meaning that represents the real edge-to-edge

margin between different classes in the training set. GICO Loss uses global information as the feedback information to optimise the intra-class and inter-class variance. To enable the calculation of GICO Loss, an algorithm is proposed to learn the cosine similarity between the class centre and the class edge. We conduct extensive experiments to evaluate these loss functions. Experimental results demonstrate their state-of-the-art performance. Furthermore, a complete face search framework for image/video has been proposed and evaluated in Chapter VI.

CHAPTER I

Introduction

1.1 Overview of Face Recognition

Biometric recognition (or biometrics) refers to the automatic recognition of individuals based on their physiological or behavioural characteristics [63]. Examples of biometric recognition include, but are not limited to, palm vein recognition, palmprint recognition, face recognition, DNA recognition, iris recognition, fingerprint recognition and retina recognition. As a type of biometric recognition, face recognition is the process of identifying or verifying individuals based on their facial features. Face recognition has some special advantages compared with other biometric recognition [62]. These advantages include: 1) contactless and passive capture; 2) strong tracking ability; and 3) cheap data acquisition devices. These advantages mean that the faces of individuals can be recognised and tracked without their conscious cooperation, and facial features can be captured with commonly used simple webcams [62].

Because of the advantages of face recognition, it has drawn a great deal of attention since its emergence, which also leads to a wide range of applications, such as smart surveillance, transaction payments and human-computer interaction:

(1) Smart surveillance. Traditional surveillance systems require security personnel to observe multiple monitoring screens simultaneously. With this approach, problems

emerge: a) vast human resource consumption; b) the security personnel's status may affect the accuracy of monitoring; c) when an emergent event happens, observers tend to neglect other events. By contrast, smart surveillance embodies computer-assisted recognition, which automatically processes and analyses a sea of video data.

(2) Transaction payments. According to the face verification results on LFW dataset [58], the accuracy of state-of-the-art methods has reached 99.77% [85]. This makes it possible to pay reliably for transactions by verifying facial features instead of using traditional methods like cash and card payment. In this way, payment efficiency will be significantly improved. Customers will not have to worry about forgetting their password, losing cash or losing bank cards.

(3) Human-computer interaction. Traditional human-computer interaction works by mouse, keyboard and monitor. A face recognition system can provide a smarter human-computer interface by recognising and analysing facial details, for example, assigning different permissions to different subjects by identity authentication; operating computers by detecting users' eye movements; perceiving users' feelings by expression recognition.

Commercially successful face recognition solutions include the solutions from IntelliVision [2], ontotext [3] and Panasonic [4]. All of their solutions can detect faces, log faces and recognise the faces that are enrolled in the database or present in a preloaded watch list in real-time. Moreover, all of them support anti-spoofing (liveness test) and can search for faces across multiple cameras. Differently, IntelliVision claims that their face recogniser has achieved an accuracy as high as 99.6% on public databases, comparable to Google and Facebook. The solution from ontotext has the ability to review the video material and locate persons unknown to the surveillance system, and can maintain a centralized database with known offenders and apply it in all chain store locations. Panasonic's solution – FacePRO, can identify faces under some difficult situations include reading faces

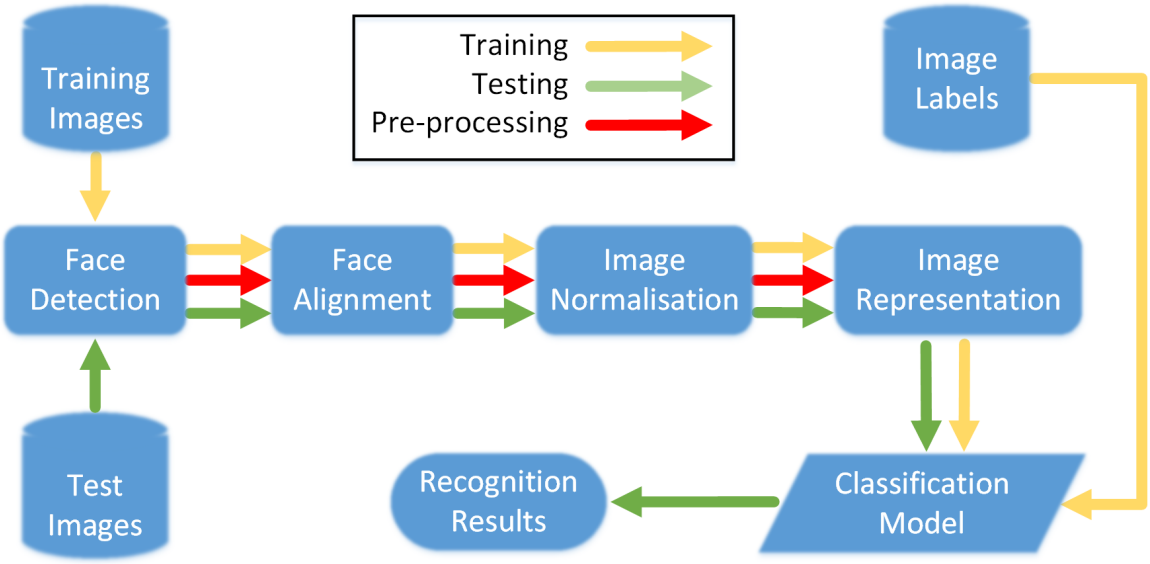


Figure 1.1: The basic framework of a face recognition system.

at an angle of up to 45 degrees to the left or right or 30 degrees up or down, with a 90% accuracy rate when detecting faces partially hidden by sunglasses or face masks.

To apply face recognition techniques to different fields, a face recognition system should be built. A typical face recognition system can recognise a face from a static image or from some video frames. These face images come from some image acquisition devices, like digital cameras and thermal imaging sensors. Fig. 1.1 illustrates the basic framework of a face recognition system which consists of three major parts.

a) **Preprocessing.** During image acquisition, image quality can be influenced by a variety of factors, such as illumination, noise and pose, which may seriously affect the recognition accuracy. Therefore, it is necessary to preprocess the images in order to eliminate the negative effects of these factors. In the raw images, faces occupy only part of the whole image, which makes face detection a necessary step before further processing. At the stage of face detection, different types of features (like histogram features, colour features, template features, structure features and Harr features) may be used to locate the accurate position of a face [53, 54, 126]. The most classic and

commonly used face detector was proposed by Viola and Jones [156], which takes advantage of Harr features and an intermediate representation for images to accelerate feature computing. Nevertheless, the detected facial areas always have some differences in pose and scale. To remove the influence of these differences, face alignment is an indispensable step in this case. A typical face alignment algorithm is based on some landmarks (e.g., the centre of the eyes, the tip of the nose and the centre of the mouth), and operates by adjusting a deformable model until the corresponding landmarks from different faces are well aligned. Active Shape Model (ASM) [27] and Active Appearance Model (AAM) [26] are two commonly used face alignment algorithms. Image normalisation aims at processing each image area to weaken the effects introduced by different illumination conditions and imaging conditions, thereby achieving some uniformity for further utilisation [152, 45].

b) Training. At this stage, training samples are first be preprocessed by face detection, face alignment and image normalisation. The preprocessed training images are then processed through a specific image representation method. With image representation, the original face image space is transformed to a facial feature space. In other words, discriminant features should be extracted from the images to describe different faces. Ideal features should be robust to various variations, like noise, illumination variance and expression. Classic hand-crafted image representation methods include Principal Component Analysis (PCA) [154, 33, 64], Linear Discriminant Analysis (LDA) [16, 95, 102], Scale-invariant Feature Transform (SIFT) [94], Local Binary Patterns (LBP) [9, 192, 82] and Gabor filters [107, 117, 93]. After image representation, the features obtained from the training samples are applied to build a classification model. As a certain type of data structure, this classification model contains the pattern information for calculating the similarities between different samples. Typical classification models include Support Vector Machine (SVM) [163], Sparse Representation [155], Neural Networks [125] and Bayesian Networks [87].

c) **Testing.** In the testing process, the face images may come from different sources, for example, real-time video frames and the test images from the test set. The raw test images also need to be preprocessed first. Then the features of the test images are extracted by a specific image representation method. The resulting features are then inputted into a trained classification model for further classification to get the final recognition results. For the large-scale testing images from the test set, the resulting recognition results can be analysed to evaluate and improve the recognition algorithm; for real-time video frames, the testing process is essential for verifying the validity and practicability of the whole system.

Face recognition tasks have two aspects — face identification and face verification. Face identification is to identify the ID of a face, answering with the exact ID, which is a multi-classification problem. Face verification aims at verifying whether two faces are from the same person, answering 'Yes' or 'No', which is a binary classification problem. The ultimate goal of face recognition research is to construct a face recognition system that is able to identify or verify a human face accurately and reliably even under uncontrolled and harsh conditions.

1.2 Research Motivation and Contributions

People are fascinated by face recognition, but the effectiveness of face recognition can be easily influenced by intrinsic and extrinsic factors such as illumination, pose, occlusion and low resolution, which makes it a challenging task to achieve a high recognition accuracy.

The existing face recognition methods are still less accurate when they are compared with some other biometric recognition methods, like iris recognition, palm vein recognition, palmprint recognition and fingerprint recognition. For example, the Equal Error Rate (EER) of a palmprint recognition method – DOC [40], can be as low as 0.009% on PolyU

3-D palmprint dataset [39] for palmprint verification task, while the lowest EER reported on IJB-C face dataset for face verification task is 1% reported by ArcFace [29]. Moreover, ArcFace needs 5.8 million images for training while DOC, as a hand-crafted method, does not need any data for training. As a result, face recognition is unable to replace other biometric recognition methods in many application scenarios and unable to perform tasks that are critically dependent on accuracy. Therefore, it is necessary to further improve the accuracy of face recognition.

Face representation plays a vital role in a face recognition system, which directly determines the recognition performance of a system. A bad face representation method will lead to the lack of discriminative ability of features. In this case, even with the best classifier, it cannot fundamentally improve the performance of face recognition. One way to improve face recognition is by face representation. The existing face representation methods can be divided into two categories: traditional face representation and deep learning-based face representation. Traditional face representation includes traditional hand-crafted representation and shallow learning-based representation, which are discussed in detail in Chapter II. Traditional face representation does not require a massive amount of training data and can be enhanced by feature fusion techniques. Deep learning-based face representation requires much more training data, but much evidence has shown that deep learning-based methods have a significant advantage on performance. The progress of deep learning on face recognition is mainly due to three important aspects — stronger network structure, larger face datasets and better loss functions. A loss function evaluates how well an algorithm models dataset and gives a loss value based on the inaccurate extent of predictions. The goal of a deep neural network is to minimize the expected loss. In a deep neural network, a loss function supervises the entire training process and provides feedback information to optimize the parameters of the neural network. Better loss functions can bring better network models and higher training speed.

Based on the consideration above, a number of original contributions have been made

in the fields of face recognition and deep learning. With our contributions, the performance of face recognition is improved, which will help enrich the application of face recognition in various fields and allow people to have a better experience in life. Detailed contributions are outlined below:

– To generate hyper-high dimensional descriptor features and highly discriminative features, a feature fusion method called Multi-descriptor Fusion (MDF) is proposed. The face identification performance of MDF is competitive compared with the state-of-the-art methods. To reduce the memory and computational costs of MDF, a novel optimisation method called Discriminant Ability-based Multi-descriptor Selection (DAMS) is proposed, which aims to find a specific number of descriptors from the entire descriptor set while maximising the discriminant ability.

– A new supervision signal is designed to improve the Centre Loss [173] and the Softmax Loss. The new supervision signal is called Minkowski Distance-based Centre Loss (MC Loss), which replaces the Euclidean distance measurement by the Minkowski distance. In this study, evaluation is conducted on two benchmark datasets – Labeled Faces in the Wild (LFW) [58] and YouTube Faces (YTF) [178]. Results show that the proposed method achieves higher verification accuracy than the Softmax Loss and the Softmax Loss + Centre Loss, and also achieves competitive results compared with the state-of-the-art methods.

– The existing loss functions, including Softmax Loss, Centre Loss and MC Loss, do not take the margin bias problem into account. To rectify this margin bias, we propose to set a minimum margin for all pairs of classes, and then design a loss function called Minimum Margin Loss (MML) based on the minimum margin. To prove the effectiveness of the proposed method, experiments are conducted on seven public datasets. Results show that MML achieved better performance than Softmax Loss, Centre Loss [173], Range Loss [196] and Marginal Loss [30], with almost no

increase in computing cost. It also achieved competitive performance compared with the state-of-the-art methods.

– MC Loss and MML are both Euclidean distance-based losses. However, in the past two years, Cosine similarity-based losses have gradually shown better performance and greater potential than Euclidean distance-based losses. Therefore, we conducted research on Cosine similarity-based losses and propose the Precise Adjacent Margin Loss (PAM Loss) to improve the discriminative ability of the deep features. To the best of our knowledge, PAM Loss is the first loss that gives 'margin' a meaning that represents the real margin between different classes in the training set. To make PAM Loss possible, we propose a learning algorithm to obtain the range of each class (namely, the cosine similarity between the class centre and the class edge). We propose and implement two versions of PAM Loss and analyse their performance variation on multiple datasets. Experimental results verified the state-of-the-art performance of PAM Loss.

– To further merge the advantages of different types of loss functions, a novel loss function (GICO Loss) is proposed. To the best of our knowledge, it is the first attempt to use global information as the feedback information. To enable the calculation of GICO Loss, we propose an algorithm to learn the cosine similarity between the class centre and the class edge. We conduct extensive experiments on five public benchmark datasets which demonstrate the state-of-the-art performance of GICO Loss.

– To make up for the shortcomings of the existing face search frameworks and apply the completed research results to practice, a new face search framework is developed for image and video retrieval. We carefully design the structure of this framework and integrate the latest face detection and face representation techniques into this framework. The evaluation of the framework shows that it can meet the needs of users.

1.3 Outline of the Thesis

The thesis is organised into seven chapters, as follows:

- Chapter I is the introduction. In this chapter, we have briefly introduced the challenges in face recognition, the advantages of face recognition, the typical structure of a face recognition system, our research motivation, our contribution in this study and the outline of the thesis.

- Chapter II provides a review of the image representation methods commonly used in face recognition including hand-crafted face representation, shallow learning-based representation and deep learning-based representation. For the first two types of representation, their representative methods are introduced. For deep learning-based representation, we introduce it by three parts — overview of deep neural networks, network architectures and loss functions.

- Chapter III proposes a multi-descriptor fusion method for face identification, which consists of two parts — initial feature extraction and descriptor selection. The experimental results on two public datasets are presented. Also, stability and runtime evaluations are provided.

- Chapter IV proposes two Euclidean distance-based loss functions for deep face recognition. The first loss function extends the Centre Loss to a Minkowki Distance-based variant, while the second loss functions aims at maximising the margin between classes. Extensive experiments are conducted and discussed.

- Chapter V proposes two Cosine similarity-based loss functions for deep face recognition. The chapter firstly introduces the proposed Precise Adjacent Margin Loss (PAM Loss), then introduces the proposed Global Information-based Cosine Optimal Loss (GICO Loss). This chapter presents how PAM Loss learns the real margin between the different classes and how the GICO Loss improves the discriminative

ability of features. Finally, the proposed methods are compared with related methods and state-of-the-art methods.

- Chapter VI presents a complete face search framework for image and video retrieval. The chapter explains the structure of this framework and the detailed techniques used in each module. An evaluation of the presented framework is also provided.

- Chapter VII summarises the key points of the main chapters, draws the overall conclusions and discusses future work.

CHAPTER II

Face Image Representation

2.1 Introduction

Face image representation is the process of extracting features from facial images for face-based machine learning tasks. In a face recognition system, face image representation is the key component. Without an excellent representation method, even the best face matching algorithm cannot recognise accurately. In the real world, face images from the same person can be quite different due to various factors including image noise, illumination variance, facial expression, facial occlusion and age. Good image representation should be robust to these variations, but it is still an unsolved and challenging task.

This section reviews some classic and widely used face image representation methods. These methods are divided into three categories: hand-crafted face representation, shallow learning-based representation and deep learning-based representation. Hand-crafted face representation is presented in Section 2.2, which refers to feature extraction based on fixed rules, where the rules are designed by researchers based on their practice, summary, observation, analysis and intuition. Section 2.3 and Section 2.4 focus on shallow learning-based representation and deep learning-based representation, respectively. Without human intervention, learning-based representation utilises automatic learning algorithms to learn a definite representation method. Compared with shallow learning, deep learning has more complicated learning targets and has many more parameters to learn. Therefore, the model

learned by deep learning usually has stronger representation ability and requires a longer time for training.

2.2 Hand-crafted Face Representation

At the early stage of face image representation, researchers proposed various hand-crafted representation methods, such as Local Binary Patterns (LBP) [108], Scale Invariant Feature Transform (SIFT) [94], Histogram of Oriented Gradient (HOG) [28], Gabor Wavelets [77] and Hidden Markov Model (HMM)[19]. This section introduces LBP, HOG and HMM, which are commonly used in face recognition.

2.2.1 Local Binary Patterns and Its Variants

LBP is an image descriptor that is originally used to describe the local texture feature of an image [109]. For each pixel of an image, the original LBP compares the base value of the pixel with the values of 8 pixels around it. The result of the comparison is an LBP value (or LBP pattern) for the pixel, which is a binary vector of length 8. The method for obtaining LBP values can be described as follows:

$$LBP = \sum_{i=0}^{7} f(g_i - g_c)2^i \tag{2.1}$$

$$f(x) = \begin{cases} 1, & \text{if } x \geqslant 0 \\ 0, & \text{if } x < 0 \end{cases} \tag{2.2}$$

where i is the index of one of the 8 surrounding pixels, g_i denotes the value of the ith pixel, g_c represents the value of the centre pixel. Therefore an LBP pattern is obtained for each pixel. All LBP patterns are counted within the entire image, resulting in a histogram over the LBP patterns. This histogram is transformed into a vector, the LBP vector, representing the image under consideration.

The original LBP has some drawbacks: considering only 8 surrounding pixels of a centre pixel, and being sensitive to image rotation and image scaling. Therefore Ojala *et al.* [109] proposed *Circular LBP*, a variant that allows for a different radius (R) and a different number of neighbours (P). This, however, introduced another problem: as the number of neighbours increases, the number of LBP patterns increases exponentially. So, there may be too many LBP patterns if more neighbours are considered. Then the LBP feature space will become too sparse to express the image texture well. *Uniform LBP* [109] is proposed to solve this problem. In Uniform LBP all binary strings of each pattern are seen as linked with the head to the tail. Moreover, those patterns that have no more than two $0/1$ transitions, called *uniform patterns*, are treated separately and are put into different bins for histogram statistics. However, all those patterns that have more than two $0/1$ transitions, called *non-uniform patterns*, are put into one bin for histogram statistics.

2.2.2 Histogram of Oriented Gradients

The original HOG is a descriptor introduced by Dalal *et al.* [28] for human detection. Moreover, as one of the most successful and popular vector-form features, HOG is inspired by SIFT [94] but different from SIFT. HOG can be regarded as a dense version of SIFT [113]. Briefly, HOG takes five steps to extract the features (See Figure 2.1):

(1) compute the gradients;

(2) divide an image into cells and blocks;

(3) build a histogram of orientation, (the number of orientations is decided by the parameter *number of bins*);

(4) normalise histograms in every block;

(5) create the HOG feature vector.

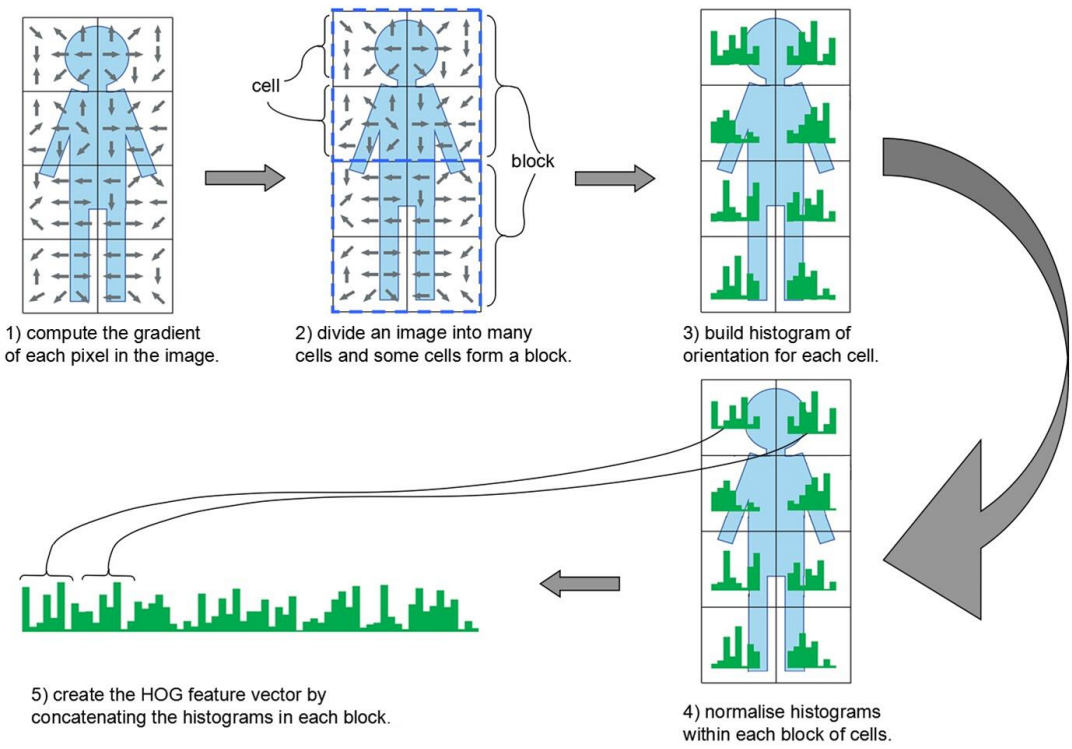


Figure 2.1: Overview of HOG. [169]

Compared with other descriptors, HOG is excellent at representing local appearance and structural objects. Nevertheless, HOG can result in high dimensional feature vectors, which leads to high storage cost and computing cost. And HOG will perform poorly if the object to be described exhibits substantial structural variation, namely the direction of the object is consistently different.

2.2.3 Hidden Markov Model

Hidden Markov Model (HMM) [19] is a prevalent statistical model, in which the system being modelled is assumed to be a Markov process. In the early days, HMM was used to process contiguous one-dimensional signals, like speech and natural language. Later on, HMM was extended to process multi-dimensional signals, including 2D images and 3D images. For a 2D image, like a face image, an image-wide rectangular window is used to sample the face image from top to bottom. As a cross-cut on some key parts (like eyes and nose) will destroy the contextual information of feature vectors, the adjacent sampling

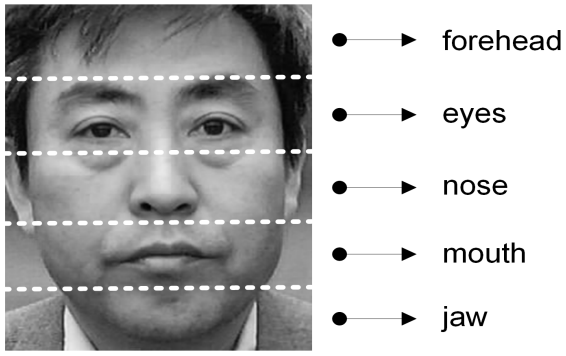


Figure 2.2: Five-state HMM for face recognition.

matrices are set to be overlapped. In general, the larger the overlapped area set, the longer the feature vector extracted, and the higher the recognition rate obtained. As shown in Fig. 2.2, the whole face image is divided into five nonoverlapped portions (i.e. jaw, mouth, nose, eyes and forehead) to achieve a Five-state HMM.

To accurately describe the image features in the rectangular windows, different methods are presented. Samaria *et al.* [129] use the grey value of each pixel directly to build the feature vector. By contrast, Nefian *et al.* [104] employ 2-dimensional Discrete Cosine Transformation (DCT), which dramatically decreases the computational complexity and the sensitivity to noise, illumination and pose.

HMM-based methods have two advantages: 1) invariance to expression and head rotation; 2) when new samples are added in, we only need to process the new samples instead of retraining using the whole training set. However, the effectiveness of this type of methods still depends on the feature extraction quality within the range of rectangular windows. Thus, how to conduct feature extraction and control the computational complexity is the key issue for HMM-based methods.

2.3 Shallow Learning-based Representation

In this section, three kinds of shallow learning-based representation methods are introduced. They are Subspace Learning, Sparse Coding and Shallow Neural network.

2.3.1 Subspace Learning

Subspace-based methods have an important assumption, namely, a face image contains much redundant information, which is negative for classification and enhances the computational cost. Based on this assumption, subspace-based methods try to look for a low dimensional subspace. The original images are projected to this subspace by linear or non-linear transforms as compact representations of the originals. Consequently, the computational cost for face recognition in this subspace is significantly reduced. Linear subspace-based methods include Principal Component Analysis (PCA) [137], Linear Discriminant Analysis (LDA) [24], and Independent Component Analysis (ICA) [13]; nonlinear subspace-based methods include Kernel Principal Component Analysis (KPCA) [70] and Kernel Discriminant Analysis (KDA) [88]. Two common linear subspace-based methods – PCA and LDA are briefly introduced below.

The basic idea of PCA comes from K-L transform. To reserve the maximum information entropy, PCA processes the training samples and extracts a group of values of linearly uncorrelated variables by orthogonal transformation. These linearly uncorrelated variables are called principal components, and their number is smaller than the dimensionality of the original samples. The original samples can be approximately represented by the linear combination of the principal components. Therefore, the purpose of dimensionality reduction is achieved. PCA has three main disadvantages: 1) the effectiveness of this method depends on the correlation between training samples and testing samples; 2) some preprocessing must be done in advance, such as normalisation; 3) lack of expansibility, as all the training samples are used for retraining when new samples are added in.

LDA (Linear Discriminant Analysis) attempts to seek a projection that can transform the original data into another feature subspace, maximising the between-class distance and minimising the within-class distance. Therefore, the overlapped areas between different classes can be minimised, which is essentially helpful for classification. Compared with PCA, LDA also has some similar disadvantages: 1) some pre-processing must be done in

advance; 2) lack of expansibility.

2.3.2 Sparse Coding

Sparse Coding has a profound theoretical basis in both biology and mathematics. In 1959, Hubel *et al.* [61] conducted a study on the simple cells of visual streak cortex of cats and found that the receptive field of neurons in the V1 region of the primary visual cortex can generate a sparse representation of visual perception information. In 1996, Olshausen *et al.* [111] published their founding in Nature that the basis functions obtained by sparse coding on the natural images have the response characteristics similar to the simple cell receptive fields in the V1 region. In 2006, Tao *et al.* [20] mathematically proved that under the condition of satisfying the Restricted Isometry Property (RIP), the sparse optimisation problems have the same solutions under the ℓ_0-norm and the ℓ_1-norm constraints. Since then, the sparse coding theory had been guaranteed in both biological and mathematical theories.

Sparse coding performs a sparse linear representation $x = \sum_{i=1}^{k} \alpha_i \phi_i$ of the input data x by learning a set of overcomplete bases, where k is the number of base vectors, α_i is the ith coding vector of x over the ith base vector ϕ_i. The sparseness of the vector means that α_i only has a few elements are non-zero, and the other elements that are close to zero. The objective function of sparse coding is as follows:

$$\begin{gathered} \arg\min_{\alpha_i^{(j)}, \phi_i} \sum_{j=1}^{N} \left\| x^{(j)} - \sum_{i=1}^{k} \alpha_i^{(j)} \phi_i \right\|^2 + \lambda \sum_{i=1}^{k} S(\alpha_i^{(j)}) \\ subject\ to\ \|\phi_i\|^2 \leq C, \forall i = 1, 2, 3, ..., k \end{gathered} \tag{2.3}$$

where $\left\| x^{(j)} - \sum_{i=1}^{k} \alpha_i^{(j)} \phi_i \right\|^2$ is the reconstruction error, $S(.)$ is the penalty to ensure sparsity, N is the number of data samples, λ controls the trade-off between the sparsity and the reconstruction error and constant C (default value is 1) is used to constrain ϕ_i so that its atoms would not reach arbitrarily high values allowing for arbitrarily low (but non-zero)

values of $\alpha_i^{(j)}$. $S(.)$ can be the ℓ_1-norm of $\alpha_i^{(j)}$ or other forms in different applications.

2.3.3 BoW Model

The Bag-of-Words (BoW) model is initially applied in text classification. However, in evaluations of combining the BoW model and Sparse Coding [72, 71], Sparse Coding was found empirically to outperform other coding methods on the object recognition tasks. Thereafter, the combination of the Bag-of-Words model and Sparse Coding was widely applied to face recognition. The BoW-based classification methods have five basic components: base descriptor, dictionary training, feature coding, feature pooling and final classification.

Wright *et al.* [179] propose Sparse Representation Classifier (SRC). SRC directly uses the training images to form a dictionary, represents each probe image (i.e. an image that is to be classified) as a linear combination of the training images, and optimises this linear combination to minimise the residual. SRC is useful for dealing with local facial occlusion, but it is mediocre at handling continuous occlusion. Therefore, numerous extensions have been proposed, for example, Structured Sparse Error Coding (SSEC) [84], Regularised Robust Coding (RRC) [188] and Robust Kernel Representation with Statistical Local Features (SLF-RKR) [187]. These extensions significantly enhanced the performance of SRC on face occlusion, but they also decreased the recognition accuracy of SRC on non-occluded data. So Huang *et al.* [59] proposed Kernel Extended Dictionary (KED), which combines Kernel Discriminant Analysis (KDA) and SRC. It was shown that KED achieved impressive results on both occluded data and non-occluded data while using fewer dictionary atoms compared with other similar methods, like Extended Sparse Representation-based Classifier (ESRC) and Superposed Sparse Representation Classifier (SSRC) [32].

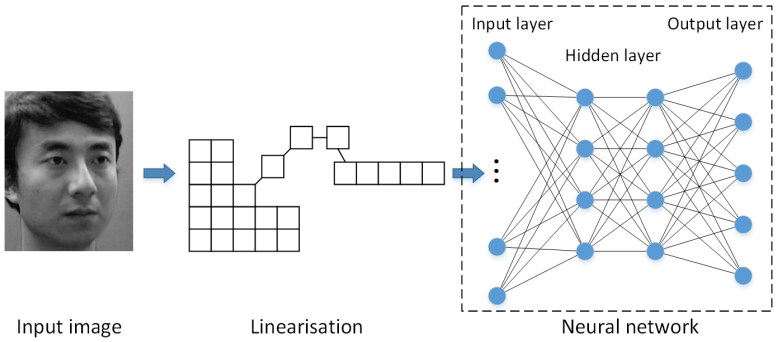


Figure 2.3: The basic structure of SNN-based representation.

2.3.4 Shallow Neural Networks

Shallow Neural network (SNN) is an important branch in face recognition, which can be used in face representation as well as classification. Meng *et al.* [38] utilise PCA for feature extraction, then employ SNN to classify the extracted feature vectors; Dong *et al.* [193] use SNN directly to extract features from the original images, and then use SNN to recognise the features obtained. The basic unit of an SNN is a 'neurone'. SNN consists of some interconnected nodes (or 'neurones') and only one or two hidden layers. A node can affect another node if there is a connection between them, while the influence efficacy is subject to the connection weight. The basic structure of SNN-based methods is shown in Fig. 2.3. According to different network structures and connection weights, neural networks can be divided into many types, such as Principal Component Neural Network (PCNN) [67], Radial Basis Function Network (RBFN) [193] and others [96, 101, 197].

Shallow Neural network has some merits, like self-organised learning, self-adaptivity, large scale distributed processing, fault-tolerance and non-linear classification. However, the feature expression ability of SNN is very limited due to insufficient middle layers, and the performance of SNN depends on a large sample size for adjusting the connection weights, which sometimes is unavailable in the real world.

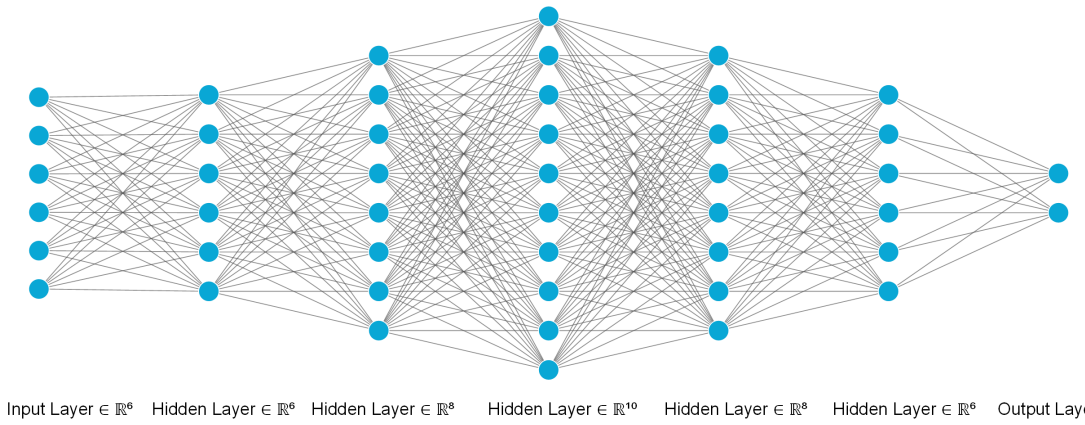


Figure 2.4: A simple deep neural network.

2.4 Deep Learning-based Representation

In this section, we firstly introduce the origin of deep neural networks in 2.4.1, then review the evolution of network architectures in 2.4.2 and finally discuss the widely used loss functions in 2.4.3.

2.4.1 Deep Neural Networks

Due to the limitation of feature expression ability, shallow learning-based representation is not ideal in complex face recognition tasks. Inspired by the visual cognition systems of mammalian brains, researchers build deeper learning frameworks for representation learning. In the 1980s, multi-layer perceptrons have been proposed. Hinton *et al.* [128] proposed the Back Propagation (BP) algorithm in 1986, which greatly reduced the training difficulties of multi-layer perceptrons. However, due to the gradient dispersion problem of BP algorithms and the lack of computing capability and training data, the research on multi-layer perceptrons had not made much progress. Until 2006, Hinton *et al.* [52] proposed the concept of deep learning and the layer-wise pre-training method, which solved the problem of gradient dispersion of deep neural networks.

Different from the shallow neural network, a deep neural network has multiple hid-

den layers of units between the input and output layers [176]. A simple deep neural network is shown in Fig. 2.4. Additional layers support the ability to compose features from lower layers to upper layers, making the potential for modelling more complex data [176]. The most common deep network architectures include stacked networks and Convolutional Neural Networks (CNNs) [81], where CNN-based architectures are the mainstream in face recognition. Stacked networks are mainly divided into stacked self-encoding networks (Stacked Autoencoder) [17] and Deep Belief Network (DBN) [52]. The main difference between them is the deep structure, where Stacked Autoencoder uses a self-encoding network and DBN uses a restricted Boltzmann machine. Deep Learning or Deep neural networks (DNNs) have been successfully used in many fields, especially in computer vision [55]. A complete DNN has two essential components — network architecture and loss function. Besides the datasets, the progress on face recognition is particularly remarkable due largely to these two important aspects.

2.4.2 Network Architectures

Compared with other network architectures, the CNNs-based architectures showed the best performance in face recognition over the past decade. Categorized by structure, these architectures include backbone networks and assembled networks. A backbone network is usually used as a whole, while an assembled network adopts backbone networks as basic blocks to meet some specific requirements such as multiple inputs or multiple tasks.

The past decade has seen some influential backbone networks including AlexNet [74], VGGNet [136], GoogLeNet [146], ResNet [50], SENet [56] and SKNet [83]. In 2012, Alex, Hinton *et al.* [74] proposed AlexNet, which achieved the best performance in the ImageNet Large Scale Visual Recognition Challenge 2012 (ILSVRC2012). AlexNet contains five convolutional layers and three fully connected layers (FC layers). Some of the convolutional layers are followed by max-pooling layers. AlexNet also integrates various techniques, such as dropout, ReLU active function and data augmentation. Two years later,

Simonyan *et al.* [136] proposed VGGNet. Like AlexNet, VGGNet also uses convolutional layer, max-pooling layer, FC layers and ReLU active function to build the network. Different from AlexNet, VGGNet has a deeper architecture (16 to 19 weight layers), which can learn progressive non-linear mappings. In 2015, Szegedy *et al.* proposed GoogLeNet [146] which introduced 'inception module' as the basic reusable building block. The inception module has a very sparse structure (namely, fewer connections) and is based on several small convolutions. Hence, the inception module has far fewer parameters, allowing it to be used to build a deeper architecture without causing over-fitting and a dramatic increase of computational resources. With the inception module, GoogLeNet consisted of 22 layers but reduced the number of parameters from 138 million (VGGNet) to 4 million and achieved better classification performance than VGGNet.

As the number of network layers increased, the problem of vanishing gradients became more and more serious. To solve this problem, Kaiming *et al.* [50] proposed Residual Neural Network (ResNet) in 2016, which utilises skip connections (or short-cuts) to jump over some layers, namely allowing the connections between those non-adjacent layers. Compared with VGGNet, ResNet increased the depth to 152 layers and showed a lower error rate in the ILSVRC competition. SENet [56] and SKNet [83] are presented in 2018 and 2019, respectively, which represent the current state-of-the-art. SENet proposed Squeeze and Excitation block (SE block) while SKNet proposed Selective Kernel unit (SK unit). Both of them can be conveniently embedded into the existing network architectures, such as ResNet, to improve their recognition accuracy. The existing CNNs only consider extracting the spacial information of images, based on which SE block also extracts the channel-wise information by explicitly modelling interdependencies between channels. SE block imposes a very limited increase in model complexity but produces significant performance improvements. The neurons in the visual cortex have self-adaptive receptive field sizes. Inspired by this, SKNet [83] allows multiple kernels to have different kernel size based on input information. Selective Kernel (SK) unit is designed to achieve this target, where

any given feature map is followed by multiple branches with different kernel sizes. These branches are then fused by the softmax-attention manner which is guided by the information in these branches. Compared with SENet, SKNet can be regarded as an enhanced version of SENet.

Assembled networks include multi-input networks and multi-task learning networks. The multi-input networks proposed in [85, 36, 199] take multiple cropped face patches as the inputs and each network branch is in charge of learning the representation of a specific facial area. The multi-input networks proposed in [167, 98, 65] take the face images of different poses as the input, where each network branch just handles a specific pose. Multi-task learning networks [121, 115] usually take identity recognition as the main task but also execute other tasks, like gender recognition, expression estimation and age estimation. The different tasks in the multi-task learning networks usually share the lower layers of the networks to learn the basic features, then split to multiple branches to achieve different tasks. For example, Ranjan *et al.* [121] propose a CNN which can simultaneously locate landmarks, detect faces, estimate pose angles, estimate age, determine smile expression and estimate gender. These tasks share the lower part of the network but are then assigned to different branches to generate the task-specific outputs.

The above-mentioned network architectures are all designed by hand. Nevertheless, designing an appropriate network architecture is a difficult and time-consuming task. In most cases, researchers have to test all kinds of architectures to search for the proper one. In the past few years, a lot of works have been done to automate the architecture search process, which makes Neural Architecture Search (NAS) be one of the hot spots in the field of Automatic Machine Learning (Auto-ML). The reinforcement learning algorithms proposed by Zoph et al. [201] and Baker et al. [12] show that good architectures can be found automatically, which marks the beginning of this new area. Soon after, Real et al. [123] demonstrated that well-studied methods in neuroevolution [42] can also find good architectures. However, all these NAS methods require massive calculation time in their

respective calculations. Therefore, many subsequent works [122, 165] have focused on simultaneously reducing the computational cost of the NAS method and searching for a network structure with high classification accuracy.

2.4.3 Loss Functions

In a classification problem, a loss function represents the price paid for inaccurate prediction. As a result, the goal of the learning problem is to minimise the expected loss. In deep neural networks, a loss function plays an extremely important role. It supervises the entire training process and provides feedback information to optimise the parameters of the neural network. A better loss function can bring better network model and higher training speed. With the rapid development of deep learning techniques, a number of loss functions are proposed to improve the performance of deep neural networks on different computer vision tasks. These loss functions fall into two categories according to their distance (or similarity) measurements: (a). Euclidean distance-based losses, and (b). Cosine similarity-based losses.

Euclidean distance-based losses. These losses include Contrastive Loss [139], Triplet Loss [131], Centre Loss [173] and Range Loss [196]. The main target of these methods is offering effective supervision signals for learning discriminative features. By inputting the positive sample pairs and negative sample pairs, Contrastive Loss tries to minimise the Euclidean distance of the positive pairs and conversely penalise the negative pairs which own small distances. The input of the Triplet Loss consists of a positive sample, a negative sample and an anchor[1]. Triplet Loss aims at pulling together the positive-anchor pairs while pushing apart the negative-anchor pairs. Nevertheless, how to generate the sample pairs is quite tricky for both Contrastive Loss and Triplet Loss. Centre Loss and Range Loss added another penalty to Softmax Loss so as to achieve joint supervision. Specifically, Centre Loss simultaneously learns the class centres and adds a penalty based on the distances be-

[1]Anchor is also a positive sample, which may be initially closer to some negative samples than it is to some positive samples.

tween the within-class samples and the corresponding class centres. Range Loss calculates the distance of the samples in each class and selects two samples with the maximum distance as the internal constraints. At the same time, Range Loss compares the distances of all class centre pairs and penalises the pair with minimum distance smaller than the specified threshold. However, it is not comprehensive to only penalise one centre pair each time, since more central pairs may have distances smaller than the specified threshold.

Cosine similarity-based losses. These losses include L-Softmax Loss [90], A-Softmax Loss [89] and AM-Softmax Loss [160]. To transform a Euclidean distance-based loss function to a Cosine similarity-based one, L-Softmax changes the formulation of the final fully-connected layer (FC layer) from $W_{y_i}^T f_i + b_{y_i}$ to $\|W_{y_i}\|\|f_i\|cos\theta_{y_i}$ by setting the bias b_{y_i} to 0, where $f_i \in R^d$ is the feature vector of the ith sample belonging to the y_ith class, $W_{y_i} \in R^d$ is the y_ith column of the weight matrix W in the final fully-connected layer, b_{y_i} is the bias term of the y_ith class and θ_{y_i} is the angle between W_{y_i} and f_i. To enlarge the angular margins between different classes, L-Softmax also adds multiplicative angular constraints to $cos\theta_{y_i}$, transforming $cos\theta_{y_i}$ to $cos(m\theta_{y_i})$, where m is a parameter called angular margin. Based on L-Softmax Loss, A-Softmax applies L2 weight normalisation, so $\|W_{y_i}\|\|f_i\|cos\theta_{y_i}$ is simplified as $\|f_i\|cos\theta_{y_i}$. With L2 weight normalisation, A-Softmax helps CNNs to learn features with geometrically interpretable angular margin. The experiments in [89] demonstrate that L2 weight normalisation improves the performance, though the improvement is very limited. AM-Softmax replaces the multiplicative angular constraints with the additive angular constraints, namely, transforming $cos(m\theta_{y_i})$ to $cos\theta_{y_i} - m$. AM-Softmax also applies feature normalisation and introduces the global scaling factor $s = 30$, which makes $\|f_i\| = s$. Hence, the training target $\|f_i\|cos(m\theta_{y_i})$ is again simplified to $s \cdot (cos\theta_{y_i} - m)$. It is worthy to note that feature normalisation brings advantages, including better performance and better geometrical interpretation, which are demonstrated in [162, 120, 91, 92].

2.5 Summary

Chapter II provided a review of the image representation methods commonly used in face recognition in the order of method evolution. These methods include hand-crafted face representation, shallow learning-based representation and deep learning-based representation. We have introduced the basic principle of these methods. Also, we have introduced the characteristics of these methods. We introduced deep learning-based representation from two aspects: the network architectures and the loss functions, as deep learning-based representation represents the current state-of-the-art.

Traditional face representation does not require a massive amount of training data and does not have high computing cost, which is now appropriate only for face identification tasks targeting a limited number of individuals. With large number of individuals, deep learning-based methods perform much better on both face identification and face verification tasks. Traditional face representation can be further enhanced by feature fusion techniques. The progress of deep learning-based representation is mainly due to stronger network structure, larger face datasets and better loss functions. With consideration to the reviewed literature, this thesis presents our contribution to traditional face representation by enhancing the performance with feature fusion, and our contribution to the deep learning-based face representation by improving the performance with new loss functions.

CHAPTER III

Selective Multi-descriptor Fusion for Face Identification

3.1 Introduction

Over the last two decades, face recognition has been an active field of research in computer vision and pattern recognition. Face verification, one form of face recognition that is to verify whether two images are of the same person, has achieved excellent performance in recent years that is better than achieved by humans. According to the latest experimental results on the benchmark LFW dataset [58], the state-of-the-art methods have achieved over 99.50% in accuracy (e.g. DeepID3 – 99.53% [140], FaceNet – 99.63% [132] and Baidu – 99.77% [86]), which are above human performance – 97.53% [76].

Face identification, another form of face recognition, is to identify the ID of a person, which is however still an unsolved problem. For the same methods, face identification is usually less accurate than face verification [80]. In the case of close-set face identification on LFW, the accuracy of DeepID3 [140] and Baidu [86] drop to 98.03% and 96.00%, respectively. Moreover, the accuracy of Baidu drops further to 92.09% if it is an open-set identification task on LFW [86]. Therefore, as an important branch of face recognition, face identification is still a challenging task.

Similarly to face verification, face identification has two critical components – face representation and face classification. The state-of-the-art methods for face representation mainly include fused descriptors [180, 170, 22] and deep learning-based methods

[148, 141, 37]. In recent years, deep learning-based methods have shown excellent performance; however, they usually need a large amount of data for training. By contrast, fused descriptors are also competitive [60, 35], especially when there is limited data available.

The remainder of this chapter is organised as follows. In Section 3.2, some related works are reviewed from three aspects – image descriptor, feature fusion method and classification method. In Section 3.3, firstly we introduce the proposed initial face representation method – MDF; then the optimisation method – DAMS is presented in detail. The experimental results on two commonly used datasets are given in Section 3.4. Finally, we summarise our method and the results in Section 3.5.

3.2 Related Work

In the field of computer vision, an image descriptor is the description of the visual features in an image or video [175]. These visual features can be shape, colour, texture, movement or other abstract features. Among a variety of image descriptors, Local Binary Patterns (LBP) [9, 118, 69] is a popular choice and has been studied extensively in the face recognition literature. LBP represents images on the basis of the grey-value differences between neighbouring pixels, which is quite effective in face identification and robust to illumination variance. Some further image descriptors have been proposed in recent years, including Chain Code-based Local Descriptor (CCBLD) [66], Discriminative Embedding Method based on the Image-to-Class Distance (I2CDDE) [198], Quaternionic Local Ranking Binary Pattern (QLRBP) [78] and Feature Descriptor using Entropy Rate (FDER) [185]. Different from LBP, which is created based on fixed sampling points in a rectangular or circular region, CCBLD builds a chain by repeatedly searching the maximum or the minimum neighbour around the current position. As a fully supervised local descriptor learning algorithm, I2CDDE tries to learn compact but highly discriminative local feature descriptors based on the image-to-class distances. QLRBP combines Quaternionic Ranking and LBP, and proposes a new quaternionic ranking function to determine

the order of two colour pixels. Different from the above-mentioned descriptors, FDER uses a graph structure to describe the image patches generated by the nonsubsampled Contourlet transform, and applies the entropy rate of random walks on the graph to build the final descriptor. In summary, different descriptors manifest in various forms, but all of them are targeted at obtaining highly-discriminative features and invariant features.

In addition, as an assistive technique, feature fusion can alleviate the unreliability brought about by using a single set of features and can introduce additional discriminant information. Feature fusion for image representation can be done at different levels by fusing either the same type of descriptors or different types of descriptors [97]. When image descriptors are combined with a feature fusion technique, they form the so-called fused descriptors. In [106], Nikan *et al.* proposed a method using feature fusion at the decision level. It divides each facial image into $M * N$ blocks, then uses Local Phase Quantisation (LPQ) and Multiscale LBP for feature extraction. In [44], Gao *et al.* fused local features and global features extracted by Gabor Wavelet Transforms (GWT) and Discrete Cosine Transforms (DCT). These features are fused through a weighted sum at the feature level. Instead of fusing features from different image descriptors, Wei *et al.* [170] fused features from the same LBP image descriptors obtained with different parameters. The base value in computing an LBP pattern is changed from the intensity of a pixel to the mean intensity of a neighbourhood of the pixel, thus the resulting LBP pattern is less sensitive to noise. Multiple LBP feature vectors are constructed at different spatial scales and combined into a weighted distance function. However, the identification accuracies using such representation methods are usually not the state-of-the-art, due possibly to the lack of sufficient discriminant information in those descriptors. To enhance the discriminative ability of fused descriptors, Dual-Cross Patterns (DCP) [35] is applied as the basic encoder which aims to maximise the joint Shannon entropy; moreover, we proposed a method called Multi-descriptor Fusion (MDF) to generate the hyper-high dimensional descriptor features. After that, we try to find a strong and robust classifier for fully utilising the power of MDF.

In face classification, classification methods include C4.5 [127], Sparse Representation Classifier (SRC) [179], k-Nearest Neighbour classifier (kNN) [171], Support Vector Machine (SVM) [51], Neural Network (NN) [79], and Naive Bayes [133], among which SRC is a popular choice. SRC works by representing each probe image (i.e. an image that is to be classified) as a linear combination of gallery image samples and optimising the linear combination to minimise the residual. SRC is good for dealing with local facial occlusion (such as random pixel corruption), but it is poor at handling continuous occlusion (due to artefacts such as a hat and sunglasses). Therefore, numerous extensions have been proposed, for example, structured sparse error coding (SSEC) [84], regularised robust coding (RRC) [188], and robust kernel representation with statistical local features (SLF-RKR) [187]. These extensions significantly enhance the robustness of SRC to face occlusion, but they may overfit the occluded training images and decrease the recognition accuracy of SRC on non-occluded data [59]. So Huang *et al.* [59] proposed Kernel Extended Dictionary (KED), which combines Kernel Discriminant Analysis (KDA) and SRC. It has been shown that KED achieves impressive results on both occluded data and non-occluded data, while using fewer dictionary atoms compared with similar methods like Extended Sparse Representation-based Classifier (ESRC) [31] and Superposed Sparse Representation Classifier (SSRC) [32]. Due to the excellent performance of KED, if not specified, we take KED as the default classifier for the proposed MDF.

To reduce the memory and computational costs of MDF, we also propose a novel optimisation method called Discriminant Ability-based Multi-descriptor Selection (DAMS), which aims to find a specific number of descriptors from the entire descriptor set while maximising the discriminant ability. The new face representation, which is refined by DAMS, is called Selective Multi-descriptor Fusion (SMDF).

3.3 Selective Multi-descriptor Fusion

3.3.1 Initial Feature Extraction

In this section, the implementation details of MDF are described. As illustrated in Fig. 3.1, MDF consists of four parts. In part one, the Supervised Descent Method (SDM) proposed by Xiong at el. [183] is introduced for landmark location. Xiong at el. trained their landmark detector with 66 landmarks on MPIE [47] and LFW-a&c [130] datasets, but only evaluated their landmark detector with 49 of the 66 landmarks on RU-FACS dataset [14], as RU-FACS dataset only provides the ground truths of 49 landmarks. To have a reliable operation, we only use these 49 landmarks. Alternatively, it may also be possible to use a subset of the 49 landmarks. However, at the first stage of our method – initial feature extraction, the control of dimensionality is not our concern. Our top priority at this stage is to generate enough features that contain as much discriminative information as possible. Feature selection will be done at the next stage to reduce the dimensionality. Besides, these 49 landmarks lie in the areas of the eyes, nose, mouth, and eyebrows. These areas contain very rich features. From our experience, the locations of all these 49 landmarks are important in a facial portion. Neglecting any of these 49 landmarks may lead to a decrease in recognition accuracy. Considering the reasons above, we use all the 49 landmarks. For each face image, the 49 landmarks are located as shown in Fig. 3.2(a). Being a state-of-the-art method, SDM has very reliable performance on landmark location, which lays a good foundation for subsequent feature extraction.

In part two, DCP [35] is applied to extract the global features for all face images. For each pixel O, Dual-cross encoder is used to obtain the value of each sampling point around it, as illustrated in Fig. 3.3. Dual-cross encoder [35] includes two types of patterns, namely, DCP_1 and DCP_2. Around each pixel, there are a total of 8 sampling points distributed on two circles. The radii of these two circles are denoted by r_{in} and r_{ex}, respectively. Before the global feature extraction, each image is divided into multiple blocks (see P2

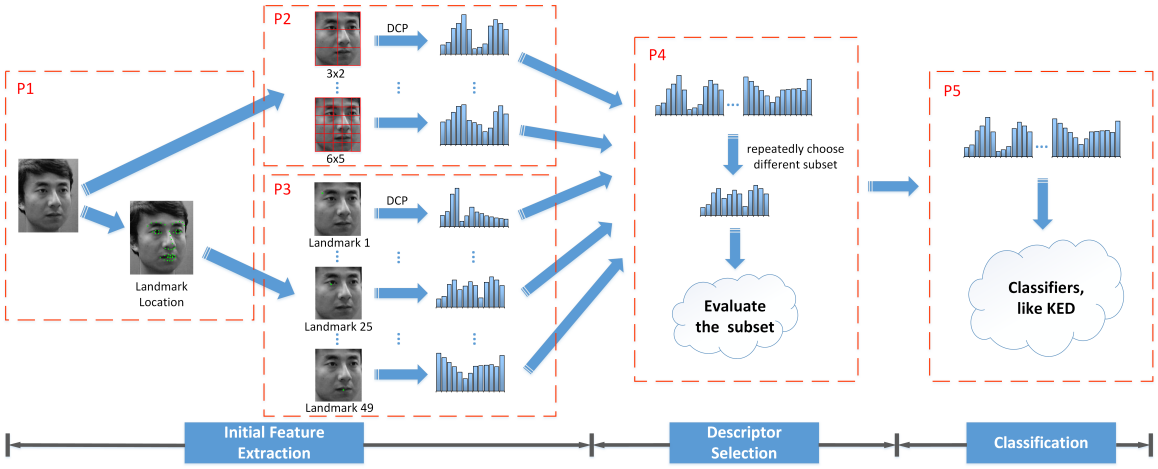


Figure 3.1: The pipeline of the proposed methods, which can be divided into three stages: initial feature extraction, descriptor selection and classification. It is worth noting that MDF includes P1, P2, P3 and P5, DAMS refers only to P4, while SMDF consists of P1 to P5.

of Fig. 3.1), which introduces two parameters – block number of each row (BNR) and block number of each column (BNC). In order to obtain sufficient global DCP features, we adjust the four variables BNR, BNC, r_{in} and r_{ex}. Then the corresponding DCP histogram is calculated for each block; the DCP histograms under the same variable combination are concatenated to form one large feature histogram. In this way, we generate 16 large feature histograms[1] as the global features for each face image. Here, choosing 16 global descriptors is mainly due to two aspects. Firstly, the combination of the global descriptors and the local descriptors requires that their total dimensionalities should be of the same order of magnitude. Ample evidence shows that only in this way can the combination be effective [150, 35, 41, 135]. Secondly, the server we use to run the experiments has a memory of 24GB, and the maximum memory usage of MDF is 22GB after applying 16

[1]Here the DCP histogram under a certain variable combination is denoted by $DCP(BNR, BNC, r_{in}, r_{ex})$. In our method, we extract the following DCP histograms for each face image: DCP(6, 5, 2, 3), DCP(6, 5, 3, 4), DCP(6, 5, 4, 5), DCP(6, 5, 5, 6), DCP(5, 4, 2, 3), DCP(5, 4, 3, 4), DCP(5, 4, 4, 5), DCP(5, 4, 5, 6), DCP(4, 4, 2, 3), DCP(4, 4, 3, 4), DCP(4, 4, 4, 5), DCP(4, 4, 5, 6), DCP(3, 2, 2, 3), DCP(3, 2, 3, 4), DCP(3, 2, 4, 5) and DCP(3, 2, 5, 6). So we get 16 DCP histograms in all for each face image. Please note that we did not carefully tune these four parameters. According to our experience, the setting of these four parameters will not significantly influence performance.

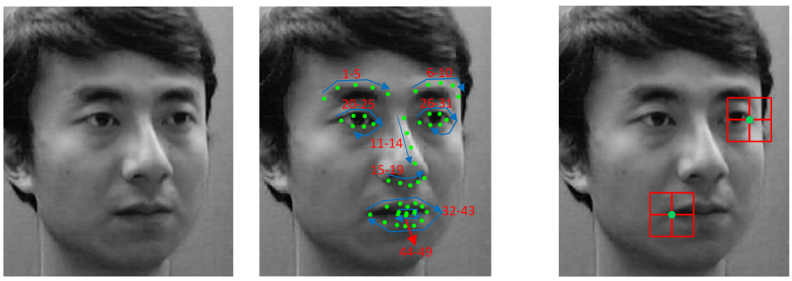


Figure 3.2: (a) Landmarks located by SDM. (b) Landmark-based local features.

global descriptors. We choose to maximise the usage of memory and generate as many global descriptors as possible so as to obtain more discriminative features. For the two reasons above, we finally choose to use 16 global descriptors.

In part three, the extraction process of the landmark-based local features is described. As illustrated in Fig. 3.2(a), 49 landmarks are located for each face image. Centred on each landmark, we define a square patch which is divided into $N * N$ non-overlapping blocks as shown in Fig. 3.2(b), where $N = 2$ in Fig. 3.2(b). One DCP histogram is calculated for each block. Hence, $N * N$ DCP histograms are calculated for each landmark. All these $N * N$ DCP histograms are concatenated to build a large histogram as the local DCP features corresponding to this landmark point. From landmark 1 to landmark 49, a total of 49 large DCP histograms are extracted for one face image. Benefiting from the maturity of facial landmark location techniques in recent years, the local features extracted in this part are robust to pose variance, expression variance and distance variance.

In part four, which is P5 in Fig. 3.1 (here, we skip P4 in Fig. 3.1, as P4 is an optional module and will be described in the next subsection), all the DCP histograms extracted in part two and part three are fused together by concatenation, and then input into classifiers for further processing. If KED is chosen, the whole dataset will be grouped into three sets, namely, training set, gallery set and testing set. Firstly, the training set is used to obtain

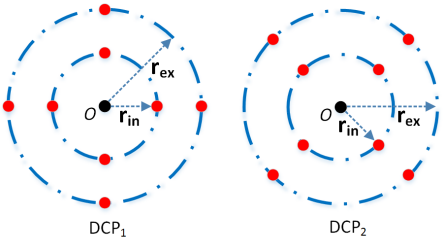


Figure 3.3: Dual-cross encoder [35] has two types of patterns – DCP_1 and DCP_2. Each pattern has eight sampling points distributed on two circles, where r_{in} and r_{ex} are the radii of the inner and exterior circles, respectively.

the $s-1$ projections by Kernel Discriminant Analysis (KDA), where s is the number of subjects (namely the number of classes). After that, the normal frontal face images and the occluded face images in the training set are applied to learn p kernel principal components which are called the occlusion model in KED. Here, p is set with the default value ten in the code exposed by the authors of KED. Please note that the subjects in the training set cannot exist in the gallery set, as the training set is used only for learning the KDA projections and the occlusion model. Then all gallery samples and occlusion model are projected by KDA projections to get the basic dictionary and extended dictionary, respectively. Finally, we use the sparse representation classifier to classify each probe image in the testing set by minimising the reconstruction residual. For more details of KED, please refer to [59].

3.3.2 Descriptor Selection

The excellent performance of MFD will be demonstrated in Section 3.4. However, it has a noticeable problem – high dimensionality, which consequently leads to high memory cost and high computational cost. Therefore, in this section, we propose a novel optimisation method called Discriminant Ability-based Multi-descriptor Selection (DAMS) to reduce the dimensionality of the feature set. The first issue that needs to be addressed is the manner of evaluating the discriminant ability. As we use Kernel Discriminant Analysis in post-processing, keeping the descriptors that can maximise the discriminant ability is a

reasonable choice. In Discriminant Analysis-based methods, the Fisher objective function is commonly used [15, 190, 25]. Thus the following Fisher objective function is initially considered to evaluate the discriminant ability of a set of descriptors:

$$J(W) = \frac{|W^T S_b W|}{|W^T S_w W|}, \tag{3.1}$$

where S_w and S_b are the within-class scatter matrix and the between-class scatter matrix, respectively.

By maximising $J(W)$, a projective matrix W^* can be found, that is:

$$W^* = \arg \max_W J(W). \tag{3.2}$$

It can be demonstrated that [143]:

$$J(W^*) = |eigV|, \tag{3.3}$$

where $eigV$ denotes the eigenvalue matrix of $S_w^{-1} S_b(n)$, which has the form:

$$eigV = \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_n \end{bmatrix}, \tag{3.4}$$

where $\lambda_1, \lambda_2, ..., \lambda_n$ denote the eigenvalues of $S_w^{-1} S_b(n)$.

Based on the settings above, DAMS (Algorithm 1) is designed to select n descriptors from the initial 65 descriptors (16 global descriptors and 49 local descriptors) according to the discriminant ability of each descriptor subset. In DAMS, we formulate an objective function called DA (Discriminant Ability) according to the specific situation in the experiments. As shown by Eq.3.5, DA is a variant of $J(W^*)$, where CFD is the Current

Algorithm 1 Select n ($n < 65$) descriptors from the whole 65 descriptors according to the discriminative ability of each descriptor subset.

Input: Feature matrix M, where M is a $i * j$ matrix, which includes i samples and each sample has j features generated by 65 descriptors. Class label vector L, a i dimensional vector, which consists of the class labels of the i samples.

Output: Index of the selected n descriptors.

Compute matrix $S_w(65)$ and matrix $S_b(65)$ based on all 65 descriptors.
Randomly select n descriptors from 65 descriptors, $D = (d_1, d_2, ..., d_n)$.
Generate $S_w(n)$ and $S_b(n)$ from $S_w(65)$ and $S_b(65)$.
Compute DA based on $S_w(n)$ and $S_b(n)$.
for $k = 1$ to 30 **do**
%Iterate for 30 times.
    for $m = 1$ to n **do**
    %Process $d_1, d_2, \ldots, d_n$ in sequence.
        for 1 to (65-n) **do**
        %Traverse all the other (65-n) descriptors.
            Replace d_m with the next one of the other (65-n) descriptors.
            Generate $S_w(n)$ and $S_b(n)$ from $S_w(65)$ and $S_b(65)$.
            Compute DA' based on $S_w(n)$ and $S_b(n)$.
            if $DA' > DA$ **then**
                $DA = DA'$
                Update the index of selected descriptors.
            else
                if $random(0, 1) < (10^{-(DA-DA')*k^{1.5}} - 0.5)$ **then**
                %Accept a worse DA at a certain probability.
                    $DA = DA'$
                    Update the index of selected descriptors.
                end if
            end if
        end for
    end for
    if $k > 20$ and DA doesn't change **then**
        return The index of selected descriptors.
    end if
end for
return The index of selected descriptors.

Feature Dimension, n is the number of descriptors in the objective subset, and 150 is the approximate mean size of all descriptor features.

$$DA = \frac{lg(\sqrt[3]{J(W^*)})}{(|CFD - 150 * n| / 100)^2 + 1} \tag{3.5}$$

$$= \frac{lg(|\sqrt[3]{eigV}|)}{(|CFD - 150 * n| / 100)^2 + 1}. \tag{3.6}$$

For the numerator of DA, we extract the cube roots of all the diagonal elements of $eigV$ as the range of $|eigV|$ is wide and mostly beyond the range of the double-precision data type. Then the lg of $|\sqrt[3]{eigV}|$ is calculated to make the numerator and denominator have the same order of magnitude. For the denominator, $(|CFD - 150 * n| / 100)^2$ is the penalty factor to balance the selection bias, as the sizes of different descriptor features are different from each other. Without this penalty factor, DA will tend to choose the descriptors that generate high-dimensional features because high-dimensional features usually have the stronger discriminant ability, which goes against our original intention – dimensionality reduction. To accelerate the penalty growth and make the numerator and denominator have the same order of magnitude, $|CFD - 150 * n|$ is divided by 100 and squared. From this penalty factor, it can be seen that the farther CFD deviates from $150 * n$ the greater the denominator and the smaller DA. When CFD equals $150 * n$, the denominator degenerates to 1 and the penalty factor loses its efficacy.

Similar to the simulated annealing algorithm, DAMS also chooses a worse DA with a certain probability, which can help DAMS to escape from a local optimum. The escape probability used in DAMS is computed as:

$$P(DA, k) = 10^{-(DA - DA') * k^{1.5}} - 0.5, \tag{3.7}$$

where DA and DA' denote the discriminant abilities obtained at different stages, and k is the current number of iterations. As k increases, $P(DA, k)$ becomes smaller. When $DA =$

DA', $P(DA, k)$ equals 0.5, which indicates that DA and DA' have the same probability of being accepted.

To accelerate the running of DAMS, two measures are taken. Firstly, the features extracted by each descriptor are processed by PCA to keep 60% of the principal component variances. Secondly, a novel method is presented for accelerating the computing of $S_w(n)$ and $S_b(n)$, where $S_w(n)$ and $S_b(n)$ are the within-class scatter matrix and between-class scatter matrix based on the selected n descriptors, respectively. In DAMS, most of the time is spent on computing $|eigV|$, where the majority of work is calculating $S_w(n)$ and $S_b(n)$ repeatedly. The new idea is to calculate $S_w(65)$ and $S_b(65)$ just for one time. After that, all the $S_w(n)$ and $S_b(n)$ ($1 \leq n \leq 65$) are generated directly from $S_w(65)$ and $S_b(65)$.

The matrices S_w and S_b are defined as below:

$$S_b = \sum_{j=1}^{s} N_j(\mu_j - \mu)(\mu_j - \mu)^T, \tag{3.8}$$

$$S_w = \sum_{j=1}^{s} \sum_{x \in X_j} (x - \mu_j)(x - \mu_j)^T, \tag{3.9}$$

where s is the class number, $N_j(j = 1, 2, \cdots, s)$ is the instance number of the jth class, $\mu_j(j = 1, 2, \cdots, s)$ is the mean vector of the jth class, and $X_j(j = 1, 2, \cdots, s)$ is the instance set of the jth class.

Let $\mu_j - \mu = \left[a_1^T a_2^T \cdots a_{65}^T\right]^T$, where μ is the mean vector of all instances and a_1, a_2, $\cdots$, a_{65} are the feature segments that correspond to the descriptor 1 to 65, respectively. We note that $a_1, a_2, \cdots, a_{65}$ are all column vectors and that they may have different sizes. Then, $(\mu_j - \mu)^T = \left[a_1^T \quad a_2^T \quad \cdots \quad a_{65}^T\right]$.

Thus, $S_b(65)$ can be denoted as

$$S_b(65) = \sum_{j=1}^{s} N_j(\mu_j - \mu)(\mu_j - \mu)^T \tag{3.10}$$

$$= \sum_{j=1}^{s} N_j \begin{bmatrix} a_1a_1^T & a_1a_2^T & \cdots & a_1a_{65}^T \\ a_2a_1^T & a_2a_2^T & \cdots & a_2a_{65}^T \\ \vdots & \vdots & \ddots & \vdots \\ a_{65}a_1^T & a_{65}a_2^T & \cdots & a_{65}a_{65}^T \end{bmatrix} \tag{3.11}$$

If we denote $\sum_{j=1}^{s} N_j a_n a_n^T = A_n A_n^T (1 \leq n \leq 65)$, then

$$S_b(65) = \begin{bmatrix} A_1A_1^T & A_1A_2^T & \cdots & A_1A_{65}^T \\ A_2A_1^T & A_2A_2^T & \cdots & A_2A_{65}^T \\ \vdots & \vdots & \ddots & \vdots \\ A_{65}A_1^T & A_{65}A_2^T & \cdots & A_{65}A_{65}^T \end{bmatrix} \tag{3.12}$$

It can be seen from (3.12) that any $S_b(n)$ can be generated from $S_b(65)$ by simply deleting the columns and rows that contain the corresponding descriptors in $S_b(65)$ but out of $S_b(n)$. For example, $S_b(6)$ involves descriptor 3 to descriptor 8, and so it can be generated by deleting the columns and rows that contain the descriptors 1, 2, 9, 10, ..., 65. As a result, we get $S_b(6)$ as follows:

$$S_b(6) = \begin{bmatrix} A_3A_3^T & A_3A_4^T & \cdots & A_3A_8^T \\ A_4A_3^T & A_4A_4^T & \cdots & A_4A_8^T \\ \vdots & \vdots & \ddots & \vdots \\ A_8A_3^T & A_8A_4^T & \cdots & A_8A_8^T \end{bmatrix} \tag{3.13}$$

It is worth mentioning that by experiment we found that DAMS spent 92.8% of the time ($n = 10$) in calculating $S_w(n)$ and $S_b(n)$ before we use the trick of block matrix operation. Thus this trick is essential in making DAMS feasible.

Figure 3.4: Examples from the CAS-PEAL-R1 dataset.

3.4 Experiments

3.4.1 Results on the CAS-PEAL-R1 Dataset

In this section, the performance of MDF and SMDF is evaluated on the CAS-PEAL-R1 dataset [43]. The CAS-PEAL-R1 dataset is constructed by the Chinese Academy of Sciences and contains 99,594 images from 1040 subjects (including 595 males and 445 females). In our experiments, we use the following subsets: 'Normal', 'Expression', 'Lighting', 'Accessory', 'Background', 'Distance' and 'Aging', which contain face images from 1040 subjects in total. Some examples from the CAS-PEAL-R1 dataset are shown in Fig. 3.4.

Following the standard experimental protocol [43], we use the whole 'Normal' subset as the gallery set; it consists of 1040 images from 1040 subjects (one sample per person). For the training set, we randomly select 400 images (100 subjects, four samples per person) from the 'Expression' subset, 800 images (200 subjects, four samples per person) from the 'Lighting' subset, 80 images (20 subjects, four samples per person) from the 'Accessory' subset; and for those subjects who appear in the above-mentioned images, we also add their images in the 'Normal' subset into the training set. Excluding the face images used in the training set, the rest of the 'Expression', 'Lighting', 'Accessory', 'Background', 'Distance' and 'Aging' subsets are used to create six probe sets respectively. The face portion of each image is cropped out and normalised to the size of 120*100 pixels. In order to ensure the veracity and reliability of our experimental results, each experiment is repeated ten times.

Here the parameter – number of descriptors is set to six in SMDF, as six descriptors are already sufficient to achieve a good result.

Comparison with SRC-based Methods. KED is an SRC-based method. To demonstrate the effectiveness of combining MDF/SMDF(6) with KED, we compare MDF/SMDF(6)+KED with other SRC-based methods, including SRC [179], ESRC [31], KDA+SRC, KDA+ESRC and KED [59]. For initial features, we use the same Multiscale LBP (MLBP) features as in [59] so as to maximise the performance of these baseline methods. Table 3.1 shows the results of different methods on three subsets. According to the results, we can observe the following:

1) All methods perform well on the Expression probe set. This is because SRC-based methods are robust to local variances such as expression and local occlusion [59].

2) MDF+KED and SMDF+KED significantly outperform other methods on the Accessory probe set; KED performs better than other methods, but it is inferior to MDF+KED and SMDF+KED. The Accessory probe set contains a large number of cases of contiguous occlusion. SRC, ESRC, KDA+SRC and KDA+ESRC fail to handle these contiguous occlusions, so they perform poorly on this probe set.

3) All methods perform relatively poorly on the Lighting probe set, but MDF+KED and SMDF+KED are still better than other SRC-based methods by at least 2.4% and 1.7%, respectively. In this case, even though the standard deviation of MDF+KED reaches 1.6, it is still acceptable.

Comparison with Other Descriptors. We also compare the proposed method with other descriptor-based methods. They are LBP [9], LTP [151], LPQ [10], POEM [157], Local Gabor XOR Patterns (LGXP) [180], Multiscale LBP [110], Multiscale tLBP (MsTLBP) [153], Multiscale dLBP (MsDLBP) [153] and DCP [35]. To maximise the performance of

Table 3.1: Comparison with SRC-based methods on Accessory, Lighting and Expression subsets of the CAS-PEAL-R1 dataset.

Method	Mean Accuracy (%) ± std. dev.		
	Accessory	Lighting	Expression
MLBP+SRC	72.9±0.6	17.3±0.7	98.2±0.4
MLBP+ESRC	87.1±1.0	82.1±0.4	99.7±0.1
MLBP+KDA+SRC	80.8±1.9	82.7±0.6	99.7±0.1
MLBP+KDA+ESRC	80.9±1.9	83.0±0.6	99.7±0.1
MLBP+KED	91.0±0.6	83.1±0.5	99.7±0.1
MDF+KED*	**97.5±0.3**	**85.5±1.6**	**99.7±0.1**
SMDF(6)+KED*	**95.2±0.5**	**85.0±1.3**	**99.4±0.2**

these baseline descriptors, we carefully choose the parameters and the distance functions (chi-squared or histogram intersection) for each of them. The final results are reported in Table 3.2 with the parameters and distance functions that can maximise the average accuracy on all subsets. According to the results, we can observe the following:

1) MDF+KED has the best identification rates on all six probe sets, which demonstrates the superiority of the proposed method.

2) MDF+KED and SMDF+KED perform much better than DCP, which demonstrates the effectiveness of the fused descriptor features extracted by MDF and SMDF.

3) The results on the Lighting and Distance probe sets indicate that the baseline methods misclassify a large proportion of the probe images on these two probe sets. An explanation is that the images from these two probe sets are significantly overlapped in the feature space, and the baseline methods have insufficient features that have strong discriminant ability.

Comparison with the State-of-the-art Methods. Finally, we compare the proposed MDF+KED and SMDF+KED with the state-of-the-art methods, including SSEC [84], RRC [188], SLF-RKR [187], MOST [124], KED and DCP. For the parameters of SSEC, they are set as in [84]: $\lambda_E = 2$, $\lambda_V = 0$, $\kappa = 0.3$, and $T = 5$. For the parameters of RRC,

Table 3.2: Comparison with other descriptors on six subsets of the CAS-PEAL-R1 dataset.

Method	Mean Recognition Accuracy (%)					
	Accessory	Lighting	Expression	Time	Background	Distance
LBP [9]	91.82	46.90	94.27	100.00	99.46	44.60
LTP [151]	91.77	47.17	94.39	100.00	99.46	44.68
LPQ [10]	92.39	57.16	93.95	100.00	99.28	44.76
POEM [157]	92.39	54.66	95.54	100.00	99.46	42.52
LGXP [180]	91.33	63.26	94.97	100.00	99.28	22.91
MsLBP [110]	92.04	47.75	95.16	100.00	99.46	44.88
MsTLBP[153]	92.74	48.06	95.41	100.00	99.46	45.48
MsDLBP [153]	90.63	48.11	92.42	100.00	99.28	37.03
DCP [35]	92.82	50.25	96.11	100.00	99.10	51.30
MDF+KED*	**97.47**	**85.49**	**99.67**	**100.00**	**99.94**	**100.00**
SMDF(6)+KED*	**95.66**	**85.09**	**99.73**	**100.00**	**99.39**	**100.00**

Table 3.3: Comparison with state-of-the-art methods on six subsets of the CAS-PEAL-R1 dataset.

Method	Source	Mean Recognition Accuracy (%)					
		Accessory	Lighting	Expression	Time	Background	Distance
SSEC [84]	TIP13'	66.6	17.4	74.5	51.9	66.8	84.2
RRC [188]	TIP13'	84.2	29.3	94.0	96.7	95.6	97.9
SLF-RKR [187]	TNNLS13'	90.9	28.8	99.6	98.5	99.9	99.7
MOST [124]	TIP14'	80.4	82.4	98.2	97.9	99.0	99.8
KED [59]	TNNLS16'	91.0	83.1	99.7	99.7	99.9	99.9
DCP [35]	TPAMI16'	92.8	50.3	96.1	100.0	99.1	51.3
MDF+KED*	N/A	**97.5**	**85.5**	**99.7**	**100.0**	**99.9**	**100.0**
SMDF(6)+KED*	N/A	**95.7**	**85.1**	**99.7**	**100.0**	**99.4**	**100.0**

Figure 3.5: Examples from the LFW dataset.

we followed the settings of [188]: $\mu = (\varsigma/\delta)$, $\varsigma = 8$, and $\tau = 0.8$. For SLF-RKR, we set $S = 0$, $P_0 = 5$, and $Q_0 = 4$ as presented in [187]. For the settings of KED and DCP, we followed [59] and [35], respectively. The comparative results are shown in Table 3.3, from which we can observe the following:

1) Compared with the state-of-the-art methods, MDF+KED and SMDF+KED still have the best identification rates on all six probe sets, which again demonstrates the good performance of the proposed methods.

2) Most methods perform well on Expression, Time and Background probe sets. However, they perform poorly on the Accessory probe set. MDF+KED achieves a better result on the Accessory probe set than other methods.

3) DCP has good performance on Accessory, Expression, Time and Background probe sets, but cannot cope well with the Lighting and Distance probe sets. KED has excellent results on all six probe sets except the Accessory probe set.

3.4.2 Results on the LFW Dataset

The LFW dataset [58] consists of more than 13,000 facial images of 5,749 subjects downloaded from the web, and has been created for research on unconstrained face recognition. The facial images in the LFW dataset have dramatic variations of illumination, occlusion, pose and expression; the only constraint is that all these faces were captured by the Viola-Jones face detector. Currently, there are four different versions of LFW, including the original version and three different types of ”aligned” versions. In the following

Table 3.4: Comparison with state-of-the-art methods on the LFW dataset.

	Method	Source	Accuracy
without outside training data	PCA600	N/A	56.9
	KDA+1NN	N/A	40.0
	KDA+SRC	N/A	89.2
	KED	N/A	89.2
	MDF	N/A	**94.3**
	SMDF(11)	N/A	**91.7**
with outside training data	COTS-s1[18]	TIFS14'	56.7
	COTS-s1+s4[18]	TIFS14'	66.5
	DeepFace[148]	CVPR14'	64.9
	WST Fusion[149]	CVPR15'	82.5
	DeepID2+[142]	CVPR15'	95.0

experiment we use the version called "LFW-a". Some original facial images from the LWF dataset are shown in Fig. 3.5. As preprocessing, all the images in LFW-a are normalised to 120*100 pixels and processed by affine transform based on three fiducial marks (left eye centre, right eye centre and mouth centre) obtained by the SDM algorithm. We use the mean value of landmarks 20 to 25 to get the position of the left eye centre, use the mean value of landmarks 26 to 31 to get the position of the right eye centre, and use the mean value of landmarks 32, 38 and 44 to 49 to get the position of the mouth centre (see Fig 3.2). With the affine transform, all the face images are aligned with the left eye centre, right eye centre and mouth centre mapped to (29, 42), (75, 42) and (53, 96), respectively.

To demonstrate the effectiveness of MDF and SMDF, we compare the proposed methods with PCA600 (reduce to 600 dimensions by PCA), KDA+1NN, KDA+SRC and KED. Different from the settings used on the CAS-PEAL-R1 dataset, the Cosine KNN classifier is used for the proposed MDF and SMDF in this section. We explored a number of classifiers and Cosine KNN shows the best performance in this case. Following the experimental protocol in [59] and [187], a subset of LFW is used in the experiments, which contains 5425 images of 311 subjects with no fewer than six samples per subject. The parameters of these methods are the same as the settings in Subsection 3.4.1. To avoid overfitting, 5-fold cross-validation is applied to all of the above methods.

Additionally, we also include comparable results on the same data by deep learning-based methods – COTS-s1 [18], COTS-s1+s4 [18], WST Fusion [149], DeepFace [148], and DeepID2+ [142]. Worthy of noting is that DeepFace and DeepID2+ are two representative deep learning-based methods for face recognition. The experimental results of different methods are presented in Table 3.4, arranged into two categories, according to the experimental routine of LFW, namely the methods with outside training data[2] and the methods without outside training data.

1) Most methods do not achieve good results on the LFW dataset because of its unconstrained and dramatic variations.

2) MDF and SMDF significantly outperform the other methods without outside data, while the identification accuracy of MDF is slightly higher than SMDF.

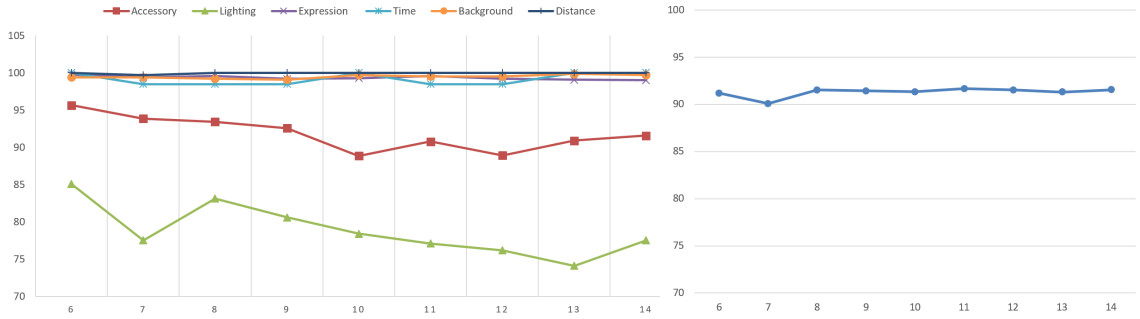
3) Compared with deep learning-based methods, MDF and SMDF are still competitive. From the perspective of identification accuracy, MDF and SMDF are better than DeepFace and WST Fusion, but a little worse than DeepID2+.

3.4.3 Stability and Runtime Evaluation

In this section, the stability of DAMS and the runtimes of the proposed methods are discussed. Firstly, we explore the relationship between identification accuracy and the number of descriptors. The number of descriptors is an important parameter in DAMS, which determines the number of target descriptors, leading to different initial feature dimension and different identification accuracy. Therefore, the following experiments were conducted on the CAS-PEAL-R1 and LFW datasets based on the same experimental settings in Section 3.4.1 and Section 3.4.2. In this process, we run DAMS for five times based on different numbers of descriptors and select the descriptor subsets that can maximise the objective

[2]In LFW dataset, "outside training data" is defined as the data that is not part of LFW [57]. As the outside training data can have a significant impact on experiments, researchers are asked to be specific about whether or what type of outside training data was used to ensure a fair comparison of different methods on LFW [57].

function. As shown in Fig. 3.6, the identification accuracies on LFW and most subsets of CAS-PEAL-R1 vary little as the number of descriptors changes from 6 to 14. However, the accuracies on Accessory and Lighting subsets show a slow downward trend. A reasonable explanation is that the proposed objective function – DA specifies the number of descriptors, but it does not specify the target dimension (it only uses a penalty factor to balance the selection bias). Whereas global descriptors have higher dimensions than the local descriptors, they are more helpful in enhancing the value of DA. So DAMS tends to choose more global descriptors rather than local descriptors as the number of descriptors increases. However, local features can cope better with the Accessory and Lighting subsets, which include different types of occlusion and illumination variations. In summary, the number of descriptors is not a sensitive parameter, but it is worth selecting a value carefully when handling some specific situations like occlusion and illumination variations.



(a) Acc. VS number of descriptors on CAS-PEAL-R1 (b) Acc. VS number of descriptors on LFW

Figure 3.6: The relationship between identification accuracy and the number of descriptors on CAS-PEAL-R1 and LFW dataset.

Using a single thread with 3.47GHz CPU (Intel Xeon X5690), we conducted experiments on the LFW dataset and recorded the runtime and relevant details of the proposed methods and some other methods we implemented. Results are shown in Table 3.5. PCA600, KDA+1NN, KDA+SRC and KED have lower requirement on memory usage and runtime on training and classification, but they have much lower recognition accuracies. For example their recognition accuracies are all lower than 90%, while the proposed MDF and SMDF have accuracies of 94.3% and 91.6%, respectively. Compared with MDF,

Table 3.5: Runtime evaluation on LFW dataset.

Method	Initial Feature Dimension	Training Time (h)	Classification Time per Instance (ms)	Memory Usage (GB)	Accuracy on LFW
PCA600	17,110	0.7h	8ms	2.6GB	56.9%
KDA+1NN	17,110	0.9h	21ms	2.3GB	40.0%
KDA+SRC	17,110	0.9h	20ms	2.3GB	89.2%
KED	17,110	1h	20ms	2.4GB	89.2%
MDF	247,808	11.1 h	187ms	22.0GB	94.3%
SMDF(14)	51,200	2.7 h	40ms	6.0GB	91.6%
SMDF(8)	52,224	2.7 h	41 ms	6.2GB	91.5%
SMDF(6)	53,248	2.7 h	41ms	6.3GB	91.2%

SMDF has a much smaller feature set, which is only approximately one-fifth of the feature set of MDF. The reduction in feature dimension leads to lower computational cost and memory cost. The training time decreases from 11.1 hours to 2.7 hours, while the classification time drops from 187 milliseconds to 40 milliseconds. Another important change is in the maximum memory cost, which is reduced to only approximately 6 GB in SMDF from the 22 GB in MDF. This enables a typical modern computer with 8 GB memory to run the proposed face identification algorithm. As a compromise, we lose $2.9\% \pm 0.2\%$ accuracy, but even so, the performance of SMDF is still better than many of the state-of-the-art methods, as illustrated in Table 3.4.

3.5 Summary

To fully utilise the discriminant information and improve the discriminative ability of features, in this chapter we proposed a high-performance face image representation method – MDF, by which we achieved higher identification accuracy than the state-of-the-art methods. Further still, we propose a novel optimisation method, DAMS, which reduces the computational cost and the memory cost of MDF. Compared with MDF, the DAMS-optimised face representation, SMDF, has a much smaller feature dimension, resulting in a much lower configuration requirement. However, SMDF still achieves excellent performance compared with other state-of-the-art methods.

CHAPTER IV

Euclidean Distance-based Losses in Deep Face Recognition

This chapter describes our research on Euclidean distance-based loss functions for deep face recognition. Firstly, we introduce our research motivation about this chapter. Then, we propose two Euclidean distance-based loss functions for deep face recognition. The next chapter focuses on Cosine similarity-based loss.

4.1 Motivation

The quantity and quality of the face datasets used for training directly influence the performance of a deep neural network in face recognition. Currently, there are a few large-scale face datasets that are publicly available, for example, MS-Celeb-1M [48], VGGFace2 [21], MegaFace [68] and CASIA WebFace [191]. As shown in Table 4.1, CASIA WebFace consists of 0.5M face images; VGGFace2 contains 3M face images in total but only from 9K identities; MS-Celeb-1M and MegaFace both contain more images and more identities, and thus should have greater potential for training a better DNN model. However, both MS-Celeb-1M and MegaFace have the problem of long-tailed distribution [196], which means that a minority of people own a majority of the face images and a large number of people have very few face images. Using datasets with long-tailed distribution, the trained model tends to overfit the classes with rich samples, thus weakening the generalisation ability on the long-tailed portion [196]. Specifically, the classes with rich samples tend

Table 4.1: Statistics for recent public available large-scale face datasets.

	MS-Celeb-1M	VGGFace2	MegaFace	CASIA
#Identities	100K	9K	672K	11K
#Images	10M	3M	5M	0.5M
Avg per Person	105	323	7	47

to have a relatively large margin between their class centres; conversely, the classes with limited samples tend to have a relatively small margin between their class centres as they occupy only a small region in space and are thus easily compressed. This *margin bias* problem is due to the long-tailed class distribution, which leads to a performance drop for face recognition [196].

Besides the training set and its class distribution, another important factor affecting performance is the loss function, which directs the network to optimise its weights during the training process. The current best-performing loss functions can be divided into two types: the loss functions based on Euclidean distance and the loss functions based on Cosine similarity. This chapter focuses on Euclidean distance-based loss and presents Minkowski Distance-based Centre Loss (MC Loss) and Minimum Margin Loss (MML).

4.2 A Minkowski Distance-based Method for Deep Face Recognition

In this section, a new supervision signal is proposed to improve the Centre Loss. The new supervision signal is called Minkowski Distance-based Centre Loss (MC Loss), which replaces the distance measurement: the Squared Euclidean distance is replaced by the Minkowski distance. In this way, we strengthen the impact of the samples on the class edge while weakening the impact of the samples around the centre. With the joint supervision of Softmax Loss and MC Loss, we can obtain deep features with high discriminative power. Experiments are conducted on two benchmark datasets – Labeled Faces in the Wild (LFW) [58] and YouTube Faces (YTF) [178]. Experimental results show that the proposed method achieves higher verification accuracy than the Softmax Loss and the Softmax Loss

+ Centre Loss, and also achieves competitive results compared with the state-of-the-art methods.

4.2.1 Softmax Loss and Centre Loss

As the proposed MC Loss is used along with Softmax Loss and is based on Centre Loss, this section briefly introduces Softmax Loss and Centre Loss. Softmax Loss is the most widely used supervision signal in deep learning, whose formulation is given in Equation 4.1.

$$L_S = -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{W_{y_i}^T f_i + b_{y_i}}}{\sum_{j=1}^{K} e^{W_j^T f_i + b_j}} \tag{4.1}$$

where N is the batch size, K is the class number of a batch, $f_i \in R^d$ denotes the feature of the ith sample belonging to the y_ith class, $W_j \in R^d$ denotes the jth column of the weight matrix W in the final fully connected layer and b_j is the bias term of the jth class. In information theory, the cross entropy between two discrete probability distributions p and q with the same support $\mathcal{X}$ is defined as follows:

$$H(p, q) = -\sum_{x \in \mathcal{X}} p(x)\ \log q(x) \tag{4.2}$$

where q is the estimated probability distribution and p is the true distribution. In the case of Softmax Loss, $p(x)$ always equals 1 and $q(x)$ is represented by $\frac{e^{W_{y_i}^T f_i + b_{y_i}}}{\sum_{j=1}^{K} e^{W_j^T f_i + b_j}}$. From Eq(4.1) and Eq(4.2), we can see that Softmax Loss is essentially the average cross-entropy between the predicted label and the true label. Cross entropy quantifies the difference between two probability distributions which means Softmax Loss aims at minimising the difference between the predicted label and the true label, namely, Softmax Loss focuses only on the correctness of classification. In other words, Softmax Loss aims at separating the samples of different classes instead of learning discriminative features and enlarging the margin between neighbour classes. Such an aim is appropriate for the closed-set tasks, like most application scenarios of object recognition and behaviour recognition. But the

application scenarios of face recognition are open-set tasks in most cases, and therefore the discriminative power of the features is extremely important for a face recognition system. To enhance the discriminative power of the features learned by Softmax Loss, Wen *et al.* proposed the Centre Loss [173] to minimise the intra-class distance, which is shown as follows:

$$L_C = \frac{1}{2}\sum_{i=1}^{N} ||f_i - c_{y_i}||_2^2 \tag{4.3}$$

where c_{y_i} denotes the class centre of the y_ith class. Centre Loss penalises all the Squared Euclidean distances between the class centres and within-class samples, and supervises the training process together with the Softmax Loss:

$$\begin{aligned} L &= L_S + \lambda L_C & (4.4)\\ &= -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{W_{y_i}^T f_i + b_{y_i}}}{\sum_{j=1}^{K} e^{W_j^T f_i + b_j}} + \frac{\lambda}{2}\sum_{i=1}^{N} ||f_i - c_{y_i}||_2^2 & (4.5) \end{aligned}$$

where λ is the hyper-parameter for balancing the two loss functions. With this joint supervision, the samples of each class are more compact and the learned deep features have strong discriminative power.

4.2.2 The Proposed Minkowski Distance-based Centre Loss

Centre Loss penalises all the Squared Euclidean distances between the class centres and within-class samples. These within-class samples include the samples close to the centre as well as the samples on the edge of the class. However, the discriminative power of the deep features is mainly determined by the samples on the class edge. Thus it is reasonable to consider improving the Centre Loss by strengthening the penalty on the samples around the edge while weakening the penalty on the samples around the centre. To achieve this target, we generalise the Squared Euclidean distance to the nth power of the Minkowski distance of order n. The Minkowski distance of order n between two points is formulated

as below:

$$D\left(X,Y\right)=\left(\sum_{i=1}^{k}\left|x_i-y_i\right|^n\right)^{1/n} \tag{4.6}$$

where $X=(x_1,x_2,\ldots,x_k)$ and $Y=(y_1,y_2,\ldots,y_k)\in R^k$. Therefore the nth power of the Minkowski distance of order n is:

$$D\left(X,Y\right)=\sum_{i=1}^{k}\left|x_i-y_i\right|^n \tag{4.7}$$

It can be seen from Equation 4.7 that the Squared Euclidean distance is actually a special case ($n=2$) of the nth power of the Minkowski distance of order n. By replacing the distance measurement, the proposed Minkowski Distance-based Centre Loss (MC Loss) is formulated as follows:

$$L_M=\frac{1}{n}\sum_{i=1}^{N}||f_i-c_{y_i}||_n^n \tag{4.8}$$

where $n\in R_{>0}$. Typically, n is set to be $2,3,4,\cdots$ in Equation 4.8. Here we can observe the change of $\left(\frac{|f_{ik}-c_k|}{|f_{jk}-c_k|}\right)^n$, where c_k is the kth element of the features of a class centre, f_{ik} is the kth element of the features of a marginal sample, f_{jk} is the kth element of the features of a near-centre sample, and $|f_{ik}-c_k|>|f_{jk}-c_k|$. $\left(\frac{|f_{ik}-c_k|}{|f_{jk}-c_k|}\right)^n$ indicates the relative impact of the marginal samples versus the near-centre samples. When $n=2$, MC Loss simplifies to Centre Loss. When $n>2$, $\left(\frac{|f_{ik}-c_k|}{|f_{jk}-c_k|}\right)^n$ is larger than it was in the Centre Loss, which means the impact of the marginal samples is relatively strengthened while the impact of the near-centre samples is relatively weakened. With the increase of n, $\left(\frac{|f_{ik}-c_k|}{|f_{jk}-c_k|}\right)^n$ will increase faster and faster, indicating the disparity between these two types of samples will get wider and wider.

Similar to Centre Loss, MC Loss takes the same approach to gradually learn the class centre positions. The details of the learning procedure of the class centres can be found in [173]. MC Loss also supervises the training process together with the Softmax Loss. The

total loss is shown below:

$$L = L_S + \lambda L_M \tag{4.9}$$

$$= -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{W_{y_i}^T f_i + b_{y_i}}}{\sum_{j=1}^{K} e^{W_j^T f_i + b_j}} + \frac{\lambda}{n}\sum_{i=1}^{N} ||f_i - c_{y_i}||_n^n \tag{4.10}$$

where λ is the hyper-parameters for adjusting the impact of MC Loss. The gradients of L_M with respect to f_i can be computed by Equation 4.11, which are similar to the Centre Loss, indicating that the CNNs supervised by MC Loss is also trainable and can be optimised by standard SGD.

$$\frac{\partial L_C}{\partial f_i} = (f_i - c_{y_i})^{n-1} \tag{4.11}$$

Algorithm 2 shows the basic learning steps in the CNNs with the proposed **Softmax + MC Loss**.

Algorithm 2 Learning algorithm in the CNNs with the proposed Softmax + MC Loss.

Input: Training samples $\{f_i\}$, initialised parameters θ_C in convolution layers, parameters W in the final fully connected layer, and initialised class centres $\{c_j | j = 1, 2, ..., n\}$. Learning rate μ^t, hyperparameter λ, learning rate of the class centres α and the number of iteration $t \leftarrow 1$.

Output: The parameters θ_C.

1: **while** t $\leq$ max number of iterations **do**
2: Calculate the total loss by $L^t = L_S^t + L_M^t$.
3: Calculate the backpropagation error $\frac{\partial L^t}{\partial f_i^t}$ for each sample i by $\frac{\partial L^t}{\partial f_i^t} = \frac{\partial L_S^t}{\partial f_i^t} + \lambda\frac{\partial L_M^t}{\partial f_i^t}$.
4: Update W by $W^{t+1} = W^t - \mu^t \frac{\partial L_S^t}{\partial W^t}$.
5: Update c_j for each centre j by $c_j^{t+1} = c_j^t - \alpha\Delta c_j^t$.
6: Update θ_C by $\theta_C^{t+1} = \theta_C^t - \mu^t \sum_i^N \frac{\partial L^t}{\partial f_i^t}\frac{\partial f_i^t}{\partial \theta_C^t}$.
7: $t \leftarrow t + 1$.
8: **end while**

4.2.3 Experiments

4.2.3.1 Experiment Settings

- **Training data and data preprocessing.** VGGFace2 [21] is chosen as our training set, as it has almost no noisy samples. To guarantee the accuracy of the results, we filtered the whole training set and removed all the face images that could be overlapped with the LFW and the YTF datasets. The filtered training set consists of 3M facial images. For the convenience of the implementation, the widely used MTCNN [194] is applied to all the training data and the testing data for face alignment and face detection, with a cropping size of 160*160. For data augmentation, random horizontal flipping is performed on the training data. To improve the verification accuracy, we concatenate the features of the original testing image and its horizontally flipped counterpart.

- **Network settings and test settings.** Based on Inception-ResNet-v1 [144], we implemented the proposed method by Tensorflow [8] and trained three relevant models according to different losses: 1. Softmax Loss, 2. Softmax Loss + Centre Loss, and 3. Softmax Loss + MC Loss. These three models are trained on one GPU (GTX 1080 Ti), and we set the network parameters as follows: batch size 90, embedding size 512, weight decay 5e-4 and keep probability 0.4. The λ in Equation 4.9 is set to be 4e-5. We set the total number of iteration to be 275K, and it costs approximately 30 hours for training. The initial learning rate is 0.05 and is then divided by ten every 100K iterations. All three models use the same parameter settings except that the model 3 loads the model 2 as the pre-trained model before the training starts, as we found this approach enables the model 3 to achieve better performance.

Figure 4.1: Examples from the LFW dataset (left) and the YTF dataset (right).

4.2.3.2 Experimental Results on LFW and YTF

In this section, we evaluate the proposed method on two public benchmark datasets – LFW and YTF datasets according to the settings in Section 4.2.3.1. Some preprocessed examples from these two datasets are shown in Fig.4.1.

To compare with other deep learning-based methods on LFW, we adopt the standard experimental protocol of unrestricted with labelled outside data [57] and do the test on 6K face image pairs according to the given pair list. YTF dataset [178] includes 3,425 YouTube videos, coming from 1,595 subjects. The number of frame of these videos ranges from 48 to 6,070, with an average of 181.3 frames. Also, we adopted the standard experimental protocol of unrestricted with labelled outside data to evaluate the proposed methods on the given 5K video pairs. Table 4.2 shows the results of the proposed method and the state-of-the-art methods on the LFW and the YTF datasets, from which we can observe the following:

- With the same framework, the proposed Softmax Loss + MC Loss outperforms Softmax Loss and Softmax Loss + Centre Loss on both two datasets. On the LFW image dataset, the verification accuracy increases from 99.43% and 99.50% to 99.57%. On the YTF video dataset, the verification accuracy increases from 94.9% and 95.1% to 95.3%. This demonstrates that the proposed MC Loss is effective on both image-

Table 4.2: Verification performance of state-of-the-art methods on LFW and YTF datasets.

Methods	Source	Images	LFW(%)	YTF(%)
Deep Face [148]	CVPR14'	4M	97.35	91.4
DeepID2+ [142]	CVPR15'	N/A	99.47	93.2
Fusion [149]	CVPR15'	500M	98.37	N/A
FaceNet [131]	CVPR15'	200M	**99.63**	95.1
Baidu [86]	arXiv15'	1.3M	99.13	N/A
Centre Loss [173]	ECCV16'	0.7M	99.28	94.9
Multibatch [147]	NIPS16'	2.6M	98.20	N/A
Aug [99]	ECCV16'	0.5M	98.06	N/A
SphereFace [89]	CVPR17'	0.5M	99.42	95.0
Range Loss [196]	ICCV17'	1.5M	99.52	93.7
Softmax Loss	N/A	3M	99.43	94.9
Softmax Loss + Centre Loss	N/A	3M	99.50	95.1
Softmax Loss + MC Loss (Proposed)	N/A	3M	99.57	**95.3**

based and video-based recognition.

- As shown in Table 4.2, we also compared the proposed method with the state-of-the-art methods including DeepID2+ [142], FaceNet [131] and SphereFace [89]. The results on LFW show that the proposed method is highly competitive and has an accuracy higher than all the other methods except FaceNet [131]. However, FaceNet utilises 200M face images for training while the proposed method only uses 3M face images during training.

- For the experiments on the YTF video dataset, we compare only the first 100 frames between two videos. However, comparing the results on YTF, we find that the proposed method is still highly competitive. It outperforms all the methods in Table 4.2, which shows the advantage of the proposed method.

4.3 Minimum Margin Loss for Deep Face Recognition

The existing loss functions including the MC Loss do not take the margin bias problem into account. To rectify this margin bias, we propose to set a minimum margin for all pairs

of classes, and then design a loss function based on the minimum margin. Inspired by Softmax Loss, Centre Loss and Marginal Loss, we propose a new loss function, *Minimum Margin Loss* (MML), which aims at forcing all the class centre pairs to have a distance larger than the specified minimum margin. Different from Range Loss, MML penalises all the 'unqualified' class centre pairs instead of only penalising the centre pair that has the smallest distance. MML reuses the centre positions constantly updated by Centre Loss, and directs the training process by joint supervision with Softmax Loss and Centre Loss. To the best of our knowledge, there is no loss function which considers setting a minimum margin between the class centres. However, it is necessary to have such a constraint to rectify the margin bias introduced by class imbalance in training data. To prove the effectiveness of the proposed method, experiments are conducted on seven public datasets – Labeled Faces in the Wild (LFW) [58], Similar-looking LFW (SLLFW) [34], YouTube Faces (YTF) [178], Megaface [68], FaceScrub [105], IJB-B [174] and IJB-C [100]. Results show that MML achieved better performance than Softmax Loss, Centre Loss, Range Loss and Marginal Loss with almost no increase in computing cost. It also achieved competitive performance compared with the state-of-the-art methods.

4.3.1 Marginal Loss and Range Loss

After combining the Softmax Loss with the Centre Loss, the within-class compactness is significantly enhanced. But it is not enough to only use Softmax Loss as the inter-class constraint, as it only encourages the separability of features. So Deng *et al.* [30] proposed Marginal Loss, which also takes the approach of joint supervision with the Softmax Loss:

$$\mathcal{L} = \mathcal{L}_S + \lambda \mathcal{L}_{Mar} \tag{4.12}$$

$$\mathcal{L}_{mar} = \frac{1}{N^2 - N} \sum_{i,j,i \neq j}^{N} \left(\xi - y_{ij} \left(\theta - \left|\left| \frac{f_i}{||f_i||} - \frac{f_j}{||f_j||} \right|\right|_2^2 \right) \right)_+ \tag{4.13}$$

where f_i and f_j are the features of the ith and jth samples in a batch, respectively; $y_{ij} \in \{\pm 1\}$ indicates whether f_i and f_j belong to the same class, $(u)_+$ is defined as $max(u, 0)$, θ is the threshold to separate the positive pairs and the negative pairs, and ξ is the error margin beside the classification hyperplane.

Marginal Loss considers all the possible combinations of the sample pairs in a batch and specifies a threshold θ to constrain all these sample pairs including the positive pairs and the negative pairs. Marginal Loss forces the distances of the positive pairs to be close to the threshold θ while forcing the distances of the negative pairs to be greater than the threshold θ. But utilising the same threshold θ to constrain both the positive and negative pairs is not appropriate, as it is often the case that the two farthest samples in a class have a distance larger than the two nearest samples of the two different but closest classes. Forcibly changing this situation will make the training procedure hard to converge.

Similar to the aforementioned methods, the Range Loss proposed by Zhang *et al.* [196] also works with Softmax Loss as the supervisory signals:

$$\mathcal{L} = \mathcal{L}_S + \lambda \mathcal{L}_R \tag{4.14}$$

Different from Marginal Loss, Range Loss consists of two independent losses, namely $\mathcal{L}_{R_{intra}}$ and $\mathcal{L}_{R_{inter}}$ to calculate the intra-class loss and inter-class loss, respectively (see Eq.(4.15)).

$$\mathcal{L}_R = \alpha \mathcal{L}_{R_{intra}} + \beta \mathcal{L}_{R_{inter}} \tag{4.15}$$

where α and β are two weights for adjusting the influence of $\mathcal{L}_{R_{intra}}$ and $\mathcal{L}_{R_{inter}}$. Mathematically, $\mathcal{L}_{R_{intra}}$ and $\mathcal{L}_{R_{inter}}$ are defined as follows:

$$\mathcal{L}_{R_{intra}} = \sum_{i \subseteq K} \mathcal{L}^i_{R_{intra}} = \sum_{i \subseteq I} \frac{n}{\sum_{j=1}^{n} \frac{1}{D_{ij}}} \tag{4.16}$$

$$\mathcal{L}_{R_{inter}} = max(M - D_{Centre}, 0) \tag{4.17}$$

$$= max(M - ||\overline{x}_{\mathcal{Q}} - \overline{x}_{\mathcal{R}}||_2^2, 0) \tag{4.18}$$

where K is the class number in current batch, D_{ij} is the jth largest distance of the sample pairs in class i, D_{Centre} is the central distance of two nearest classes in current batch, $\overline{x}_{\mathcal{Q}}$ and $\overline{x}_{\mathcal{R}}$ denote the class centres of class $x_{\mathcal{Q}}$ and $x_{\mathcal{R}}$ which have the shortest central distance, and M is the margin threshold. $\mathcal{L}_{R_{intra}}$ measures all the sample pairs in a class and selects n sample pairs that have large distances to build the loss for controlling the within-class compactness. As described in [30], experiments show that $n = 2$ is the best choice. $\mathcal{L}_{R_{inter}}$ aims at forcing the class centre pair that has the smallest distance to have a larger margin up to the designated threshold. But there are more centre pairs that may have distances larger than the designated threshold. It is not comprehensive enough to consider only one centre pair each time, which leads the training procedure to take a long time to completely converge because of the low learning speed.

4.3.2 The Proposed Minimum Margin Loss

Inspired by Softmax Loss, Centre Loss and Marginal Loss, we propose the Minimum Margin Loss (MML) in this chapter. MML is used in conjunction with Softmax Loss and Centre Loss, where Centre Loss is utilised to enhance the within-class compactness, Softmax and MML are applied for improving the between-class separability. Specifically, Softmax is in charge of guaranteeing the correctness of classification while MML aims at optimising the between-class margins. The total loss is shown below:

$$\mathcal{L} = \mathcal{L}_S + \alpha\mathcal{L}_C + \beta\mathcal{L}_M \tag{4.19}$$

where α and β are the hyper-parameters for adjusting the impact of Centre Loss and MML.

MML specifies a threshold called Minimum Margin. By reusing the class centre positions updated by Centre Loss, MML filters all the class centre pairs based on the specified Minimum Margin. For those pairs which have distances smaller than the threshold, corre-

sponding penalties are added to the loss value. MML is formulated as follows:

$$\mathcal{L}_M = \sum_{i,j=1}^{K} max(||c_i - c_j||_2^2 - \mathcal{M}, 0) \tag{4.20}$$

where K is the class number of a batch, c_i and c_j denote the class centres of the ith and jth classes respectively, and $\mathcal{M}$ represents the designated minimum margin. In each training batch, the class centres are updated by Centre Loss with the following two equations:

$$c_j^{t+1} = c_j^t - \gamma \Delta c_j^t \tag{4.21}$$

$$\Delta c_j^t = \frac{\sum_{i=1}^{m} \delta(y_i = j)\dot{(}c_j - f_i)}{1 + \sum_{i=1}^{m} \delta(y_i = j)} \tag{4.22}$$

where γ is the learning rate of the class centres, t is and the iteration number and $\delta(condition)$ is a conditional function. If the condition is satisfied, $\delta(condition)$ equals 1, otherwise $\delta(condition)$ equals 0. Note that, in Range Loss, the centre of a class is computed by averaging the samples of this class in a batch. However, the size of a batch is limited, and the sample size of a particular class may be more limited. Therefore, the class centres generated in this way are not precise compared with the real class centres. Compared with Range Loss, the learned class centres of MML are closer to the real class centres.

Algorithm 3 shows the basic learning steps in the CNNs with the proposed $\mathcal{L}_S + \mathcal{L}_C + \mathcal{L}_M$.

4.3.3 Discussion

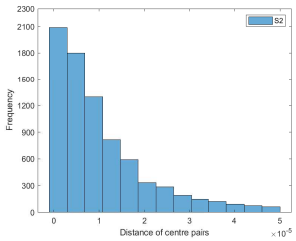
Can MML enlarge distances of the closest class centre pairs that are smaller than the specified minimum margin? To verify this point, we use the deep models trained by Scheme I (Softmax Loss + Centre Loss) and Scheme II (Softmax Loss + Centre Loss + MML) to extract the features of all the images from a cleaned version of VGGFace2 dataset [21]. The details of the cleaned dataset and the training process of these two models are presented in Section 4.3.4.1. The difference between Scheme I and Scheme II is

Algorithm 3 Learning algorithm in the CNNs with the proposed $\mathcal{L}_S + \mathcal{L}_C + \mathcal{L}_M$.

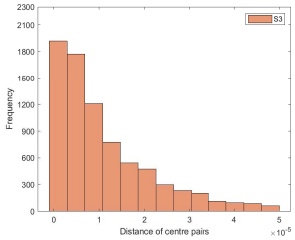
Input: Training samples $\{f_i\}$, initialised parameters θ_C in convolution layers, parameters W in the final fully connected layer, and initialised n class centres $\{c_j|j = 1, 2, ..., n\}$. Learning rate μ^t, hyperparameters α and β, learning rate of the class centres γ and the number of iteration $t \leftarrow 1$.

Output: The parameters θ_C.

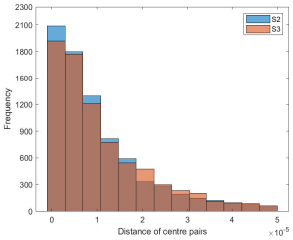
1: **while** not converge **do**
2: Calculate the total loss by $\mathcal{L}^t = \mathcal{L}_S^t + \alpha\mathcal{L}_C^t + \beta\mathcal{L}_M^t$.
3: Calculate the backpropagation error $\frac{\partial\mathcal{L}^t}{\partial f_i^t}$ for each sample i by $\frac{\partial\mathcal{L}^t}{\partial f_i^t} = \frac{\partial\mathcal{L}_S^t}{\partial f_i^t} + \alpha\frac{\partial\mathcal{L}_C^t}{\partial f_i^t} + \beta\frac{\partial\mathcal{L}_M^t}{\partial f_i^t}$.
4: Update W by $W^{t+1} = W^t - \mu^t\frac{\partial\mathcal{L}^t}{\partial W^t} = W^t - \mu^t\frac{\partial\mathcal{L}_S^t}{\partial W^t}$.
5: Update c_j for each centre j by $c_j^{t+1} = c_j^t - \gamma\Delta c_j^t$.
6: Update θ_C by $\theta_C^{t+1} = \theta_C^t - \mu^t\sum_i^N\frac{\partial\mathcal{L}^t}{\partial f_i^t}\frac{\partial f_i^t}{\partial\theta_C^t}$.
7: $t \leftarrow t + 1$.
8: **end while**



(a) Without using MML



(b) After using MML



(c) Comparison between S2 and S3

Figure 4.2: For each class in VGGFace2, its corresponding nearest neighbour class can be found by comparing the positions of different class centres. (a), (b) and (c) show the distributions of the distances between every class centre and its corresponding nearest class centre. Specifically, (a) shows the distribution in the case of using the features generated by Scheme I (without using MML). (b) shows the distribution in the case of using the features generated by Scheme II (using MML). (c) shows the comparison results of (a) and (b), where S1 and S2 represent Scheme I and Scheme II, respectively.

that Scheme II employs MML as a part of the supervision signal but Scheme I does not. With the extracted features, we calculate the centre position for each class and then calculate the distance between each class centre and its corresponding closest neighbour class centre. The distributions of the distances of these class centres are shown in Figure 4.2. Figure 4.2(a) and Figure 4.2(b) show the distance distributions of Scheme I and Scheme II, respectively. Figure 4.2(c) makes a comparison between Scheme I and Scheme II, from which we can see that Scheme II has smaller values in the first five bins but larger values in the rest of the bins. This indicates that MML enlarges the distance of some neighbour centre pairs, and therefore increases the quantity of the centre pairs having a large margin.

Can MML truly improve the performance of the model on face recognition? To answer this question, we conduct extensive experiments on different benchmark datasets as illustrated in Section 4.3.4. The experimental types include face verification, face identification, image-based recognition and video-based recognition. Results show that the proposed method can beat the baseline methods as well as some state-of-the-art methods.

4.3.4 Experiments

In this section, we describe the implementation details of the experiments, investigate the influence of the parameters β and $\mathcal{M}$, and evaluate the performance of the proposed method. The evaluations are conducted on MegaFace [68], FaceScrub [105], LFW [58], SLLFW [34], YTF [178], IJB-B [174] and IJB-C [100] datasets with face identification and face verification tasks. Face identification and face verification are two main tasks of face recognition. Face verification aims at verifying whether two faces are from the person, answering 'Yes' or 'No', which is a binary classification problem. Face identification is to identifying the ID of a face, answering the exact ID, which is a multi-classification problem.

4.3.4.1 Experiment Details

Training data. In all experiments, we use VGGFace2 [21] as our training data. To ensure the reliability and the accuracy of the experimental results, we removed all the face images that might overlap with the benchmark datasets. As the label noise in the VGGFace2 is very low, no data cleaning has been applied. The final training dataset contains 3.05M face images from 8K identities.

Data preprocessing. MTCNN [194] is applied to all the face images for landmark location, face alignment and face detection. If face detection fails on a training image, we simply discard it; if it fails on a testing image, the provided landmarks are used instead. All the training and testing images are cropped to 160*160 RGB images. To augment the training data, we also perform random horizontal flipping on the training images. To improve the recognition accuracy, we concatenate the features of the original testing image and its horizontally flipped counterpart. Note that we did not do any data cleaning on all the testing sets involved in the experiments including Megaface dataset. Doing cleaning on MegaFace may be controversial, as some researchers consider it unfair for the methods previously tested on the non-cleaned dataset[1]. According to the results published by the MegaFace team, the best methods using cleaned data can have an accuracy higher than 99% while the best method (BingMMLab-v1) using non-cleaned data has an accuracy of only 83.758%[2].

Network settings. Based on Inception-ResNet-v1 [144], we implemented and trained five models by Tensorflow [8] according to five supervision schemes: Softmax Loss, Softmax Loss + Centre Loss, Softmax Loss + Marginal Loss, Softmax Loss + Range Loss and Softmax Loss + Centre Loss + MML. For convenience, we use "Softmax Loss", "Centre Loss", "Marginal Loss", "Range Loss" and "MML" to represent these five schemes, respectively, in presenting the experimental results. We train these three models on one

[1]For example, the discussion on GitHub: https://github.com/deepinsight/insightface/issues/49

[2]Results published by MegaFace team: http://megaface.cs.washington.edu/results/facescrub.html

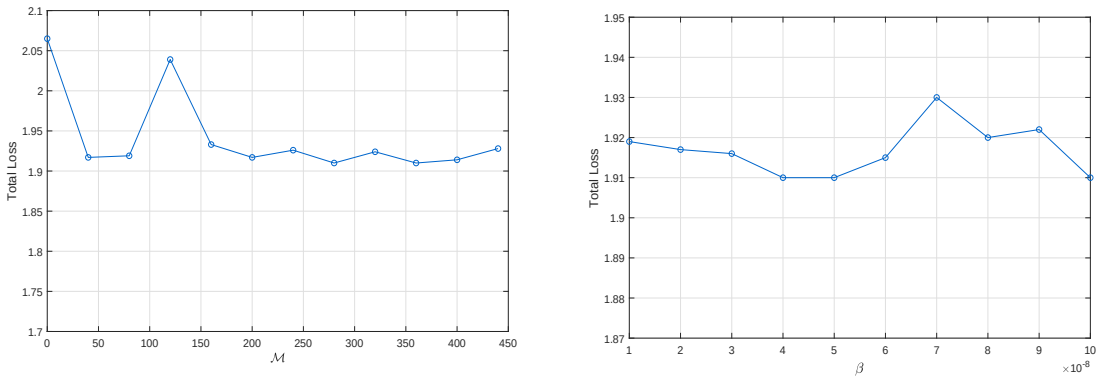
GPU (GTX 1080 Ti), and we set 90 as the batch size, 512 as the embedding size, 5e-4 as the weight decay and 0.4 as the keep probability of the fully connected layer. The total number of iterations is 275K, costing about 30 hours of computation. The learning rate is initiated as 0.05 and is divided by ten every 100K iterations. All schemes use the same parameter settings except that Softmax Loss + Centre Loss + MML loads the trained model of Softmax Loss + Centre Loss as the pre-trained model before training starts, as this approach enables the former to achieve better recognition performance. Since the training of Softmax Loss + Centre Loss finishes when it converges completely, just reloading the model and resuming training without changing any parameters will not improve the model. In training, the model needs to learn two abilities: the ability to separate different classes (making different classes have no overlap) at the first stage and the ability to enlarge the margin between different classes at the second stage. MML focuses only on the target of the second stage. In addition, MML uses the learned class centres for computing; however, the learned class centres cannot reflect the real centres at the early stage as it requires some time for learning. Applying MML at the first stage will cause interference to the training at this stage. Actually, this two-stage training mode can also be regarded as a one-time training by initialising the factor – β to 0 and then setting it to 5e-8 after a certain number of epochs. These two modes are equivalent.

Test settings. During the testing, we try our best to find the parameter settings that lead to highest performance. The α and β in Eq.(4.19) are set to be 5e-5 and 5e-8, respectively. The minimum margin of MML is set to be 280. The deep feature of each image is obtained from the output of the fully connected layer, and we concatenate the features of the original testing image and its horizontally flipped counterpart, and therefore the resulting feature size of each image is $2 * 512$. The final verification results are achieved by comparing the threshold with the Euclidean distance of two features

4.3.4.2 Influence Analysis of Parameters β and $\mathcal{M}$

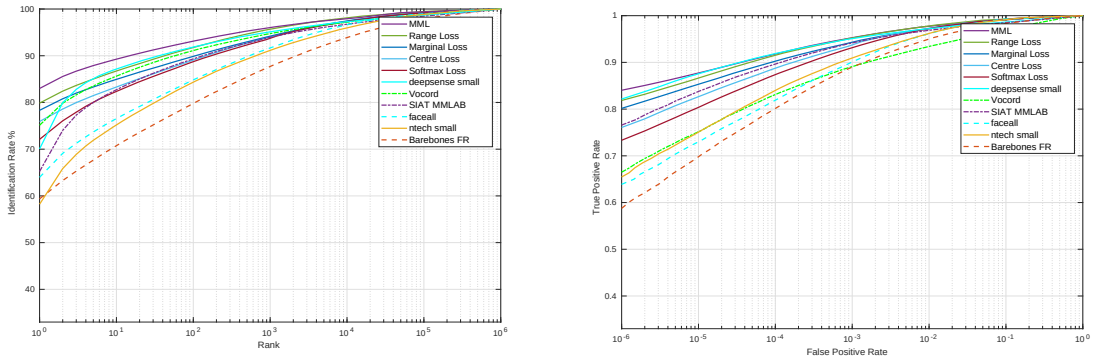
β is the hyper-parameter for adjusting the impact of MML in the combination. $\mathcal{M}$ is the designated minimum margin. These two parameters influence the performance of the proposed method. Therefore, how to set these two parameters is a question worthy of study.

Total loss only reflects the performance of the model on the training set. We conduct two experiments on VGGFace2 dataset and evaluate the influence of these two parameters on total loss. In the first experiment, we fixed β to 5e-8, and observe the influence of $\mathcal{M}$ on total loss as shown in Figure 4.3(a). In the second experiment, we fixed $\mathcal{M}$ to 280, and evaluate the relationship between β and total loss as shown in Figure 4.3(b). From Figure 4.3(a), we see that setting $\mathcal{M}$ to 0, namely without using MML, is not appropriate, as it leads to a high total loss. The lowest total loss occurs when $\mathcal{M}$ is 280. From Figure 4.3(b), we can observe that the total loss remains stable with a wide range of β, but reaches its lowest value when β is 5e-8. Therefore, in the subsequent experiments, we fixed $\mathcal{M}$ and β to 280 and 5e-8, respectively.



(a) (b)

Figure 4.3: Total loss of two groups of models: (a) fixed β = 5e-8, and different $\mathcal{M}$, (b) fixed $\mathcal{M}$ = 280, and different β.



(a) (b)

Figure 4.4: (a) reports the CMC curves of different methods with 1M distractors on MegaFace Set 1. (b) reports the ROC curves of different methods with 1M distractors on MegaFace Set 1.

Table 4.3: The identification rates and the verification rates of different methods on Megaface and FaceScrub datasets with 1M distractors. The results of the benchmark methods in the upper part of the table are generated with the features provided by the MegaFace team[4].

Methods	Rank1@10^6	Rank100@10^6	VR @FAR10^{-6}	VR @FAR 10^{-5}
Barebones FR	59.36%	79.79%	58.77%	69.80%
ntech small	58.21%	84.34%	65.48%	75.07%
faceall	63.97%	84.84%	63.89%	72.99%
SIAT MMLAB	65.23%	89.33%	76.56%	83.78%
Vocord	75.13%	91.11%	66.50%	75.15%
deepsense small	70.06%	91.85%	82.15%	87.56%
Softmax Loss	72.11%	88.73%	73.33%	80.37%
Centre Loss	75.93%	89.07%	76.07%	82.66%
Marginal Loss	78.32%	89.87%	80.16%	85.32%
Range Loss	79.86%	91.76%	81.85%	86.65%
MML	**83.00%**	**93.12%**	**84.03%**	**87.73%**

4.3.4.3 MegaFace Challenge 1 on FaceScrub

In this section, we conduct experiments with the MegaFace dataset [68] and the FaceScrub dataset [105]. The MegaFace dataset consists of a million faces and their respective bounding boxes obtained from Flickr (Yahoo's dataset). The FaceScrub dataset is a publicly available dataset containing 0.1M images from 530 identities. According to the experimental protocol of MegaFace Challenge 1, the MegaFace dataset is used as the distractor set, while the FaceScrub dataset is used as the test set. The evaluation is conducted with the officially provided code [68]. More details about the experimental protocol can be found in [68].

We compare the proposed method (MML) with different losses and some deep learning-based methods provided by the MegaFace team[5]. In the face identification experiments, the Cumulative Match Characteristics (CMC) curves [116] are calculated to measure the ranking capabilities of different methods, as illustrated by Figure 4.4(a)). In the face verification experiments, we use the Receiver Operating Characteristic (ROC) curves to evaluate the different methods. The ROC curves plot the False Acceptance Rate (FAR) of a 1:1 matcher versus the False Rejection Rate (FRR) of the matcher, which are shown in Figure 4.4(b). Table 4.3 presents the numeric results of different methods on identification rates and the verification rates with 1M distractors.

From Figure 4.4(a), Figure 4.4(b) and Table 4.3, we can observe that MML performs better than other deep learning-based methods on both identification and verification test. This demonstrates the effectiveness of the whole framework. The proposed MML consistently outperforms Softmax, Centre Loss, Marginal Loss and Range Loss, which confirms the effectiveness of the proposed loss function.

[5]The features of the benchmark methods provided by MegaFace team: (http://megaface.cs.washington.edu/participate/challenge.html)

Figure 4.5: Examples from the LFW dataset (left) and the YTF dataset (right).

Table 4.4: Verification rates of state-of-the-art methods on LFW and YTF datasets.

Methods	Source	Training Images	LFW(%)	YTF(%)
Range Loss [196]	ICCV17'	1.5M	99.52	93.7
Marginal Loss [30]	CVPR17'	4M	99.48	96.0
VGG Face [114]	BMVC15'	2.6M	98.95	97.3
Deep Face [148]	CVPR14'	4M	97.35	91.4
FaceNet [131]	ICCV15'	200M	99.63	95.1
Centre Loss [173]	ECCV16'	0.7M	99.28	94.9
Multibatch [147]	NIPS16'	2.6M	98.20	
Aug [99]	ECCV16'	0.5M	98.06	
SphereFace [89]	CVPR17'	0.5M	99.42	95.0
Contrastive CNN [49]	ECCV18'	0.5M	99.12	
OE-CNNs [168]	ECCV18'	1.7M	99.47	
Softmax Loss	N/A	3.05M	99.43	94.9
Centre Loss	N/A	3.05M	99.50	95.1
Range Loss	N/A	3.05M	99.50	95.1
Marginal Loss	N/A	3.05M	99.52	95.3
MML (Proposed)	N/A	3.05M	99.63	95.5

4.3.4.4 Comparison with the State-of-the-art Methods on LFW and YTF Datasets

In this section, we evaluate the proposed method on LFW [58] and YTF [178] datasets according to the settings in Section 4.3.4.1. We follow the standard experimental protocol of unrestricted with labelled outside data [57] on both LFW and YTF datasets. Table 4.4 shows the results of the proposed method and the state-of-the-art methods on LFW and YTF datasets, from which we can observe the following.

- The proposed MML outperforms Softmax Loss and Centre Loss, increasing the verification performance both on LFW and YTF datasets. On LFW, the accuracy improves from 99.43% and 99.50% to 99.63%, while on YTF, the accuracy increases from 94.9% and 95.1% to 95.5%. Also, MML outperforms Range Loss and Marginal Loss both on LFW and YTF datasets. On LFW, the accuracy improves from 99.50% and 99.52% to 99.63%, while on YTF, the accuracy increases from 95.1% and 95.3% to 95.5%. This demonstrates the effectiveness of the MML, and also demonstrates the effectiveness of the combination of Softmax Loss + Centre Loss + MML.

- Compared with the state-of-the-art methods, the proposed method has an accuracy of 99.63% on LFW and 95.5% on YTF, higher than most of the methods. FaceNet has similar performance to the proposed method on LFW, but FaceNet uses a large scale dataset that includes approximately 200 million face images. Consequently, FaceNet requires much more time for training than the proposed method, which only uses 3.05 million face images.

4.3.4.5 Further Comparison on SLLFW Dataset

As more and more methods are gradually touching the theoretical upper limit[6] of LFW, the performance gaps between different methods become smaller and smaller, making it hard to differentiate different methods. Therefore, to confirm the performance of MML,

[6]There are six mismatched pairs on LFW which are incorrectly labelled as matched. So the upper limit accuracy on LFW is (6000-6)/6000=99.90%.

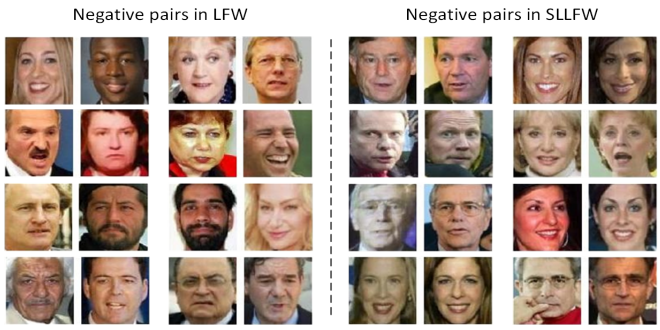


Figure 4.6: Examples of the negative pairs in LFW and SLLFW. Compared to the negative pairs in LFW, the negative pairs in SLLFW are quite difficult to distinguish.

Table 4.5: Verification performance of different methods on SLLFW.

Method	Source	Training Images	LFW(%)	SLLFW(%)
Deep Face [148]	CVPR14'	0.5M	92.87	78.78
DeepID2 [139]	NIPS14'	0.2M	95.00	78.25
VGG Face [114]	BMVC15'	2.6M	96.70	85.78
DCMN [34]	PR[J]17'	0.5M	98.03	91.00
Noisy Softmax [23]	CVPR17'	0.5M	99.18	94.50
Softmax Loss	N/A	3.05M	99.43	95.92
Centre Loss	N/A	3.05M	99.50	96.02
Range Loss	N/A	3.05M	99.50	96.07
Marginal Loss	N/A	3.05M	99.52	96.07
MML	N/A	3.05M	99.63	96.37

an additional experiment is conducted on SLLFW [34]. SLLFW uses the same positive pairs as LFW for testing, but in SLLFW, 3000 similar-looking face pairs are deliberately selected from LFW by human crowdsourcing to replace the random negative pairs in LFW. Some examples of the negative pairs in LFW and SLLFW are shown in Fig. 4.6. Compared with LFW, SLLFW adds more challenges to the testing, causing the accuracy of the same state-of-the-art methods to drop by 10-20%.

Table 4.5 shows the verification accuracy of different methods on SLLFW. The results of some benchmark methods are shown in the top half of the table. These results are publicly accessible [7] and provided by the SLLFW team[34]. As can be seen from Table 4.5, MML achieves considerably better performance than the benchmark methods on SLLFW. Also, MML shows higher accuracy than other relevant loss functions. In the top half of the

Table 4.6: Evaluation results with 1:1 verification protocol on IJB-B and IJB-C datasets.

Method	IJB-B TAR@FAR=1e-4	IJB-C TAR@FAR=1e-4
Crystal Loss [119]	0.898	0.919
ResNet50 [21]	0.784	0.825
SENet50 [21]	0.800	0.840
ResNet50+SENet50 [21]	0.800	0.841
MN-v [182]	0.818	0.852
MN-vc [182]	0.831	0.862
ResNet50+DCN(Kpts) [181]	0.850	0.867
ResNet50+DCN(Divs) [181]	0.841	0.880
SENet50+DCN(Kpts) [181]	0.846	0.874
SENet50+DCN(Divs) [181]	0.849	0.885
GAN+ArcFace [184]	0.904	0.926
PCP+ArcFace [184]	0.901	0.924
PCPSM+ArcFace [184]	0.907	0.928
LRR+ArcFace [184]	0.909	0.931
PCPSFM+ArcFace [184]	0.911	0.934
Softmax Loss	0.908	0.931
Centre Loss	0.910	0.934
Range Loss	0.916	0.937
Marginal Loss	0.917	0.939
MML	**0.921**	**0.943**

table, the accuracy of the benchmark methods drops by between 16.75% and 4.68% from LFW to SLLFW. By comparison, the accuracy of MML drops by 3.26%. The results on SLLFW further confirm the performance of the proposed methods.

4.3.4.6 Results on IJB-B and IJB-C

The IJB-B dataset [174] is composed of 21.8K still images and 55K frames from 7,011 videos. In IJB-B, there are 1,845 subjects which have no overlap with the popular face recognition benchmarks, such as VGGFace2 [21] and CASIA WebFace [191]. In IJB-B, there are 12,115 templates in total with 10,270 genuine matches and 8M impostor matches. The IJB-C dataset [100] is an extension of IJB-B. It contains 31.3K still images and 117.5K frames from 11,779 videos. All these images and videos are from 3,531 subjects which also have no overlap with the popular face recognition benchmarks. In IJB-C, there are 23,124

templates in total including 19, 557 genuine matches and 15, 639K impostor matches.

Following the 1:1 verification protocol, we compare the proposed MML with the most recent methods as shown in the upper part of Table 4.6. For a fairer comparison, we also directly compare MML with other popular and relevant loss functions under the same framework. Results show that MML performs better than the most recent methods as shown in the upper part of Table 4.6 on both IJB-B and IJB-C datasets. Also, MML shows better performance than the relevant loss functions compared in the lower part of Table 4.6.

4.4 Summary

In this chapter, we focus on Euclidean distance-based losses and explore how to improve the existing losses. To further minimise the intra-class distance, we proposed a Minkowski distance-based generalisation method for improving the Centre Loss for deep face recognition. The resulting new loss function is called MC Loss. Experiments are conducted on the LFW image dataset and the YTF video dataset. Results demonstrate the effectiveness of MC Loss for unconstrained image-based and video-based face recognition and show that the proposed method is highly competitive even compared with the state-of-the-art methods. However, the existing loss functions including the MC Loss do not take the margin bias problem into account. Therefore, we proposed Minimum Margin Loss (MML) to solve this problem. We show that MML is very easy to implement in CNNs and our CNN models can be directly optimised by the standard SGD. We compare MML with the methods published in the past few years in the leading conference and journals. We also directly compare MML with the relevant loss functions under the same framework. Results show that MML has state-of-the-art performance.

CHAPTER V

Cosine Similarity-based Losses in Deep Face Recognition

This chapter describes our research on Cosine similarity-based loss functions for deep face recognition. Firstly, we introduce our research motivation about this chapter. Then, we propose two Cosine similarity-based loss functions for deep face recognition.

5.1 Motivation

Convolutional neural networks (CNNs) have demonstrated impressive performance for face recognition, where the loss function plays an important role in this process. Softmax Loss is the most commonly used loss function in deep learning. However Softmax Loss is not the best choice in face recognition as it encourages only the separability of features instead of the discriminative ability of features. Most of the face recognition tasks are open-set tasks which require the features to have strong discriminative ability. To learn highly discriminative features, many different loss functions have been proposed in recent years [139, 131, 173, 30, 196, 90, 89, 160, 29].

The loss functions based on Cosine similarity include L-Softmax Loss [90], A-Softmax Loss [89] and AM-Softmax Loss [160]. These losses are also derived from Softmax Loss. They achieved state-of-the-art performance in deep face recognition, but they have three common defects. Firstly, the 'margin' referenced in the above losses is the margin between the decision boundaries of Softmax, which does not represent the real margin between the

different classes in the training set. Secondly, the above losses impose a margin on all possible combinations of the class pairs, which is not wise or necessary. This chapter presents Precise Adjacent Margin Loss (PAM Loss) and Global Information-based Cosine Optimal Loss (GICO Loss). To precisely optimise the real edge-to-edge margin, we propose PAM Loss in Section 5.2. As PAM Loss considers optimising only the inter-class distance, we propose GICO Loss in Section 5.3 to simultaneously optimise both inter-class and intra-class distance.

5.2 Precise Adjacent Margin Loss for Deep Face Recognition

In this section, we propose the Precise Adjacent Margin Loss (PAM Loss), which gives 'margin' a meaning that represents the real margin between the different classes in the training set. Different from the above losses, PAM Loss optimises only the margin between a limited number of class pairs.

5.2.1 Related Work

As described in Section 4.2.1, the purpose of Softmax Loss is to separate samples of different classes, instead of learning discriminative features and expanding the margin between different classes. Therefore, Softmax Loss is only suitable for closed-set tasks such as object recognition and behaviour recognition in most cases. However, it is difficult or even impossible to collect all the faces that may appear in the test phase, so most of the application scenarios of face recognition are open-set tasks, where Softmax is not a proper option.

To enhance the discriminative ability of the features, L-Softmax Loss, A-Softmax Loss, AM-Softmax Loss and ArcFace Loss are proposed successively in the past two years. Eq.(5.1) and Eq.(5.2) show the formulation of the L-Softmax Loss and the A-Softmax

Loss, respectively:

$$\mathcal{L}_L = -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{\|W_{y_i}\|\|f_i\|\psi(\theta_{y_i})}}{e^{\|W_{y_i}\|\|f_i\|\psi(\theta_{y_i})} + \sum_{j=1,j\neq y_i}^{P} e^{\|W_j\|\|f_i\|\psi(\theta_j)}} \tag{5.1}$$

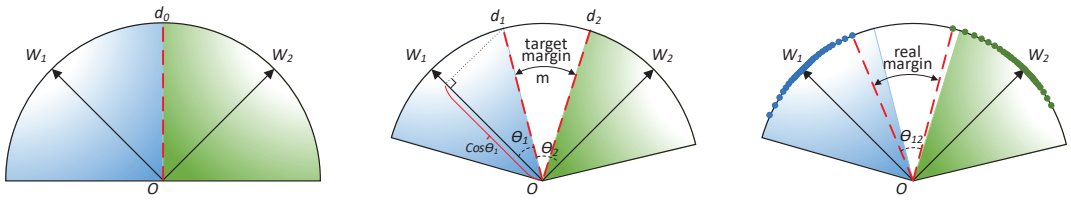
$$\mathcal{L}_A = -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{\|f_i\|\psi(\theta_{y_i})}}{e^{\|f_i\|\psi(\theta_{y_i})} + \sum_{j=1,j\neq y_i}^{P} e^{\|f_i\|\psi(\theta_j)}} \tag{5.2}$$

where N is the batch size, P denotes the class number of the entire training set, $f_i \in R^d$ is the feature of the ith sample in the mini-batch, y_i is the class label of the ith sample, $W_j \in R^d$ denotes the jth column of the weight matrix W in the final fully connected layer and $\psi(\theta_{y_i})$ equals $(-1)^k cos(m\theta_{y_i}) - 2k, \theta_{y_i} \in (\frac{k\pi}{m}, \frac{(k+1)\pi}{m})$, $k \in (0, m-1)$, $m \geq 1$ denotes the size of the angular margin, which is used to adjust the targeting angular margin. Based on the original Softmax Loss in Eq.(4.1), L-Softmax Loss and A-Softmax Loss transform the FC layer formulation from $W_{y_i}^T f_i + b_{y_i}$ to $\|W_{y_i}\|\|f_i\|cos\theta_{y_i}$ by fixing the bias b_{y_i} to 0. As a result, the distance measurement is transferred from the Euclidean distance to the cosine similarity. Different from L-Softmax Loss, L2 weight normalisation is applied in A-Softmax Loss by fixing $\|W_{y_i}\| = 1$. In L-Softmax Loss and A-Softmax Loss, m is introduced by multiplication on the θ_{y_i}, and therefore the corresponding margin is called multiplicative angular margin.

On the basis of A-Softmax, AM-Softmax adopts L2 feature normalisation and replaces $\psi(\theta_{y_i})$ with $cos(\theta_{y_i}) - m$, which is called additive cosine margin:

$$\mathcal{L}_{AM} = -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{s(cos(\theta_{y_i})-m)}}{e^{s(cos(\theta_{y_i})-m)} + \sum_{j=1,j\neq y_i}^{P} e^{scos(\theta_j)}} \tag{5.3}$$

where $\|f_i\|$ is fixed by L2 normalisation and is re-scaled to s. After AM-Softmax, Deng *et al.* proposed ArcFace Loss, which further updates $cos(\theta_{y_i}) - m$ with $cos(\theta_{y_i} + m)$. Even though ArcFace still uses the additive margin, it has better geometric meaning as m directly



(a) Softmax with Weight and Feature Normalisation. (b) Variants based on Softmax and Margin Constraint. (c) Interpretation of the Real Margin.

Figure 5.1: Geometrical interpretation of (a) Softmax Loss with weight and feature normalisation, (b) variants based on Softmax and margin constraint, and (c) the real margin. These three sub-figures are all in the case of 2D feature space.

corresponds to the angular margin. ArcFace Loss is therefore formulated as:

$$\mathcal{L}_{arc} = -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{s(cos(\theta_{y_i}+m))}}{e^{s(cos(\theta_{y_i})-m)} + \sum_{j=1,j\neq y_i}^{P} e^{scos(\theta_j)}} \tag{5.4}$$

Compared with multiplicative margin-based losses, additive margin-based losses have some advantages: (a) easy to implement; (b) fast to converge; and (c) better performance.

5.2.2 The Proposed PAM Loss

Fig. 5.1 provides a 2D visualisation of the aforementioned losses in the case of two classes. The blue area and green area represent the targeting areas of the two classes, respectively. In Fig. 5.1, W_1 and W_2 are the corresponding weights of class 1 and class 2 in the FC layer. As described in Fig. 6 of [161], W_1 and W_2 will finally converge to the centres of their corresponding classes, and therefore W_1 and W_2 can be regarded as the approximate centres of class 1 and class 2, respectively. In Fig. 5.1(a), d_0 is the decision boundary between class 1 and class 2. In Fig. 5.1(b), d_1 and d_2 are the corresponding decision boundaries of class 1 and class 2. From the formulation of the Softmax Loss and the aforementioned variants, it can be inferred that the targeting area is determined by W_j in Softmax Loss and is determined by W_j and m in the variants. The geometrical interpretation of the Softmax Loss is shown in Fig. 5.1(a), where $W_1^T d_0 = W_2^T d_0$. Therefore, class 1 and class 2 share the same decision boundary d_0 and there is no margin between

the targeting areas of class 1 and class 2. Fig. 5.1(b) illustrates the case of the variants based on Softmax and margin constraint. d_1 and d_2 are the corresponding decision boundaries of class 1 and class 2 and there is a margin between the targeting area of the two classes. The size of the margin is determined by m. For example, if the variant is AM-Softmax, $W_1^T d_1 - m = W_2^T d_1$ and $W_2^T d_2 - m = W_1^T d_2$. If the variant is ArcFace Loss, $\|W_1\|\|d_1\|cos(\theta_1 + m) = \|W_2\|\|d_1\|cos\theta_2$. Since $\|W_1\| = \|d_1\| = 1$, $cos(\theta_1 + m) = cos\theta_2$ and $m = \theta_2 - \theta_1$, which is the angle of the margin.

However, the targeting area of a class is not its real area. Its real area is determined by the samples of this class in the training set. Ideally, the real area is expected to finally converge to the targeting area. But, during the training, the real area could be smaller or bigger than the targeting area, or even has no overlap with the targeting area, which is very common at the early stage of the training. So the resulting margin actually should be called the targeting margin, which is different from the real margin. A simple example is shown in Fig. 5.1(c). In the training process, the quality of the weights in CNNs are judged by the corresponding class distribution in the training set. In other words, the feedback from the training set guides the updating of the weights in CNNs. Therefore, we think the feedback information should be as precise as possible and it is necessary to introduce the real margin into the loss.

The aforementioned variants impose a margin in all possible combinations of class pairs. This approach is simple and convenient but not wise or necessary. The first original purpose of training is to separate the overlapped classes. The second original purpose is to enlarge the margin of those classes that are close to each other. For those classes which are already far from each other, there is no need to optimise the weights to make them even farther apart, as they have satisfied the requirement of classification. And the expressive power of a neural network is not unlimited. Optimising the weights to satisfy the requirement of a part of the classes will inevitably have an influence on the distribution of other classes.

For the above reasons, we propose the Precise Adjacent Margin Loss (PAM Loss). PAM Loss is used along with the AM-Softmax Loss (see Eq. (5.5)) and has two versions whose formulations are shown in Eq. (5.6) and Eq. (5.7), respectively.

$$\mathcal{L} = \mathcal{L}_{AM} + \lambda \mathcal{L}_{PAM} \tag{5.5}$$

$$\begin{aligned} \mathcal{L}_{PAM\_v1} &= \frac{\sum_{Top}(S, P)}{P} \\ S &= \{cos(\theta_{ij}) : i, j = 1, 2, 3, ..., P; i > j\} \end{aligned} \tag{5.6}$$

$$\begin{aligned} \mathcal{L}_{PAM\_v2} &= \frac{\sum_{i=1}^{P} \sum_{Top}(S_i, 2)}{2P} \\ S_i &= \{cos(\theta_{ij}) : j = 1, 2, 3, ..., P\} \end{aligned} \tag{5.7}$$

where λ is the hyper-parameter for adjusting the impact of the loss, θ_{ij} is the real margin angle between class i and class j as illustrated in Fig. 5.1(c), S and S_i are sets of $cos(\theta_{ij})$, and $\sum_{Top}(S, K)$ denotes the sum of the K maximum elements in set S. Both versions of PAM Loss aim to optimise the margins between different classes. The ideal approach is to optimise the margins of all the adjacent classes. However, it is extremely time-consuming to select out all these adjacent classes in the hypersphere. In PAM Loss v1, a conservative strategy is adopted. PAM Loss v1 penalises P pairs of classes which have the minimum margins (P is the number of classes). This is because the minimum number of pairs of the adjacent classes is P, which occurs when all the classes line up in a circle on the surface of the hypersphere. In PAM Loss v2, we try another strategy, namely paying attention to every class. The basic idea of PAM Loss v2 is to find the nearest neighbour class of each class and penalise the margin between them.

Calculating $cos(\theta_{ij})$ is the key aspect of the PAM Loss. To calculate $cos(\theta_{ij})$, two parts are needed: class centre and cosine range of the class, where cosine range of the class means the cosine similarity between the class centre and the farthest sample of the class. As the training goes on, W_j gradually converges to the centre of the class j ($j = 1, 2, ..., P$), which has also been described in Fig. 6 of [161]. W_j is easy to obtain from the FC layer, so

W_j is used as the approximation to replace the centre of the jth class. For the cosine range of class j, we propose the following learning algorithm to recursively study and update the range of the class. $R(j)$ is initialised to 1 and is then updated iteratively with the following equations:

$$R(j)^{t+1} = R(j)^t + \sum_{i=1}^{N} \phi(y_i, j) \cdot \Delta R_i, j = 1, 2, ..., P. \tag{5.8}$$

$$\Delta R_i = \begin{cases} cos(W_{y_i}, f_i) - R(y_i)^t, & R(y_i)^t > cos(W_{y_i}, f_i) \\ \beta \cdot (cos(W_{y_i}, f_i) - R(y_i)^t), & R(y_i)^t \leq cos(W_{y_i}, f_i) \end{cases} \tag{5.9}$$

where $\phi(y_i, j) = 1$ if $y_i = j$, $\phi(y_i, j) = 0$ if $y_i \neq j$, β (called shrink rate) is used to adjust the shrink speed of the class range. Eq. (5.9) contains two aspects: (a) when the cosine similarity between the input sample and the corresponding class centre is less than the current recorded class range, the class range is directly replaced by their cosine similarity; (b) When the cosine similarity between the input sample and its corresponding class centre is greater than the recorded class range, we let the class range shrink by the product of β and their cosine similarity. Part(a) keeps the range of the class up to date, but, as the training goes on, the range of the real class range tends to become smaller and smaller. Therefore, part(b) helps the learned class range shrink to the real value.

With the class centre and the cosine range of class, the $cos(\theta_{ij})$ can be calculated. Let $R(i) = cos(\theta_i)$, $R(j) = cos(\theta_j)$, $W_i \cdot W_j = cos(\theta)$, then $cos(\theta_{ij}) = cos(\theta - \theta_i - \theta_j)$. By solving this equation, we get:

$$\begin{aligned} cos(\theta_{ij}) =& W_i W_j R(i) R(j) - W_i W_j (\sqrt{(1 - R(i)^2)(1 - R(j)^2)} \\ &+ \sqrt{(1 - (W_i W_j)^2)(1 - R(i)^2)} R(j) \\ &+ \sqrt{1 - (W_i W_j)^2} R(i) R(j). \end{aligned} \tag{5.10}$$

In our coding implementation, we do a one-time calculation to obtain all $cos(\theta_{ij})(i, j = 1, 2, 3, ..., P; i > j)$ by matrix manipulation between W and $(R(1), R(2), ..., R(P))$.

Table 5.1: Parameter settings about the network and the testing.

Parameter	**Value**	**Parameter**	**Value**
batch size	120	optimizer	ADAM
image size	160*160	weight decay	0.0005
epoch size:	1000	moving average decay	0.9999
embedding size	512	AM-Softmax scalar	40.0
random flip	True	AM-Softmax margin	0.3
keep probability	0.4	PAM Loss shrink rate	0.01
Learning Rate Schedule			
epoch 0∼99		0.05	
epoch 100∼199		0.005	
epoch 200∼360		0.0005	

5.2.3 Experiments

5.2.3.1 Implementation Details

We implement a total of four schemes with Tensorflow [8] by combining Inception-ResNet-v1 [144] with different losses: (1). ResNet+Softmax, (2). ResNet+AM-Softmax, (3). ResNet+AM-Softmax+PAM Loss v1, and (4). ResNet++AM-Softmax+PAM Loss v2. For convenience, we use 'Softmax', 'AM-Softmax', 'PAM Loss v1' and 'PAM Loss v2' to represent the above four schemes in presenting the experimental results, respectively.

VGGFace2 [21] is used as our training set in all experiments. We removed the face images in VGGFace2 that might overlap with the benchmark testing set to ensure the reliability of the experimental results. The resulting training set consists of 3.05M facial images from 8K identities. For all face datasets involved in the experiments, we apply MTCNN [194] for face detection. MTCNN fails on detection sometimes. If it fails on a training image, we just remove it from the training set. If it fails on a testing image, we use the landmarks or the bounding boxes provided by the authorities. The detailed parameter settings for the network and the testing are given in Table 5.2.3.1.

5.2.3.2 Results on LFW and YTF

In this section, we conduct the face verification test and compare the proposed PAM Loss with the state-of-the-art methods on two benchmark datasets – LFW [58] and YTF [178]. In the experiments, we follow the standard experimental protocol of unrestricted with labelled outside data [57]. Based on the protocol, we test 6,000 face pairs in LFW according to the given image list and test 5,000 video pairs according to the given video list.

Table 5.2 shows the results of the proposed methods and the state-of-the-art methods on LFW and YTF datasets, from which we can observe the following. On LFW, both versions of PAM Loss outperform the related works: Softmax, L-Softmax, A-Softmax and AM-Softmax. FaceNet has the same accuracy as the proposed PAM Loss v1. However, FaceNet uses 200 million images for training while the PAM Loss uses only 3.05 million images for training. Compared with the other state-of-the-art methods, PAM Loss has the highest verification accuracy. On YTF dataset, PAM Loss v1 has an accuracy of 96.14%, which is higher than all the other methods. PAM Loss v2 has very similar performance to Marginal Loss, but Marginal Loss uses a larger training set and has poorer performance on LFW than PAM Loss v2. Results on LFW and YTF datasets demonstrate the effectiveness and the state-of-the-art performance of the proposed methods.

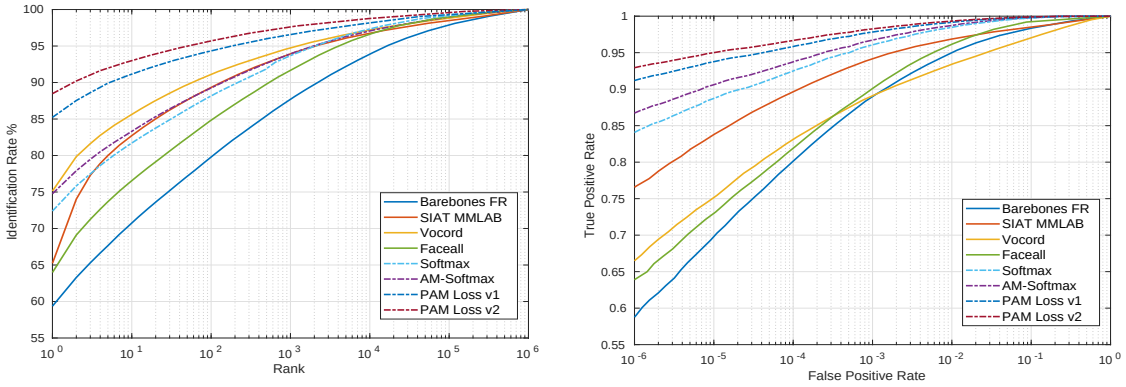
5.2.3.3 MegaFace Challenge 1 on FaceScrub

In this section, experiments are conducted on the MegaFace dataset [68] and the FaceScrub dataset [105]. We follow the experimental protocol of MegaFace Challenge 1, where MegaFace is set as the distractor set while FaceScrub is set as the testing set. The evaluation code [68] is provided by the authority. More details about the experimental protocol can be found in [68].

Fig. 5.2(a) and Fig. 5.2(b) show the CMC curves and the ROC curves with 1 Million distractors on MegaFace Set 1, respectively. The results of the benchmark methods

Table 5.2: Verification accuracy of the state-of-the-art methods on LFW and YTF datasets.

Methods	Source	Images	LFW	YTF
Range Loss [196]	ICCV17'	1.5M	99.52	93.7
Marginal Loss [30]	CVPR17'	4M	99.48	96.0
DeepID2+ [142]	CVPR15'		99.47	93.2
Deep Face [148]	CVPR14'	4M	97.35	91.4
Fusion [149]	CVPR15'	500M	98.37	
FaceNet [131]	ICCV15'	200M	99.63	95.1
Baidu [86]	arXiv15'	1.3M	99.13	
Centre Loss [173]	ECCV16'	0.7M	99.28	94.9
Multibatch [147]	NIPS16'	2.6M	98.20	
Aug [99]	ECCV16'	0.5M	98.06	
L-Softmax [90]	ICML16'	0.5M	98.71	
A-Softmax [89]	CVPR17'	0.5M	99.42	95.0
Softmax	N/A	3.05M	99.50	95.22
AM-Softmax	N/A	3.05M	99.57	95.62
PAM Loss v1	N/A	3.05M	99.63	96.14
PAM Loss v2	N/A	3.05M	99.62	96.00



(a) (b)

Figure 5.2: (a) reports the CMC curves of different methods with 1M distractors on MegaFace Set 1. (b) reports the ROC curves of different methods with 1M distractors on MegaFace Set 1.

(including Barebones FR, SIAT MMLAB, Vocord and Faceall) are generated from the features provided by the MegaFace team[1]. It can be seen from Fig. 5.2(a) and Fig. 5.2(b) that PAM Loss v1 and PAM Loss v2 have better identification and verification performance than Softmax, AM-Softmax, and other benchmark methods. PAM Loss v2 outperforms PAM Loss v1 in both figures, which indicates that PAM Loss v2 has a stronger ability in the case of 1 million distractors. The results on the MegaFace and the FaceScrub datasets confirm the effectiveness of the proposed methods.

5.3 GICO Loss for Deep Face Recognition

In this section, we propose a new loss function, namely Global Information-based Cosine Optimal Loss (GICO Loss) for face recognition. The deep model trained with GICO Loss is named GicoFace.

5.3.1 The Proposed GICO Loss

Typical Euclidean distance-based loss functions include Centre Loss [173], Marginal Loss [30] and Range Loss [196]. Details of these loss functions have been reviewed in Section 2.4.3. It is worthy to note that Marginal Loss and Range Loss follow two important targets: minimising intra-class variance and maximising inter-class variance. Centre Loss follows only the first target. From experimental results for the Euclidean distance-based loss functions [139, 131, 173, 30, 196], it can be found that both two targets of improving discriminative ability contribute to performance, which is also demonstrated in Section 5.3.5.

Table 5.3 summarises the properties of the most recent and best-performing loss functions. We can see that these loss functions either do not apply weight and feature normalisation such as Contrastive Loss, Triplet Loss, Centre Loss, Range Loss and Marginal

[1]The download link of features provided by MegaFace team: (http://megaface.cs.washington.edu/participate/challenge.html)

Loss; or do not explicitly optimise intra-class and inter-class variance, such as L-Softmax Loss, A-Softmax Loss, AM-Softmax Loss and ArcFace Loss. However, these four properties are all beneficial for learning more discriminative features. Obviously, further optimising intra-class and inter-class variance can lead to better discriminative ability. The experiments in [89] demonstrate that L2 weight normalisation improves the performance, though the improvement is very limited. Feature normalisation brings advantages, including better performance and better geometrical interpretation, which are demonstrated in [162, 120, 91, 92].

The properties of GICO Loss are also shown in Table 5.3, where it can be seen that GICO Loss possesses all four properties of optimising intra-class and inter-class variance, and weight and feature normalisation. Different from all other loss functions, GICO Loss is guided by the distribution information from the whole training set.

Table 5.3: Properties of different loss functions in deep face recognition.

	Optimise Intra-class Variance	Optimise Inter-class Variance	WN	FN	Feedback Source
Contrastive Loss [139]	Yes	Yes	No	No	mini-batch
Triplet Loss [131]	Yes	Yes	No	No	mini-batch
Centre Loss [173]	Yes	No	No	No	mini-batch
Marginal Loss [30]	Yes	Yes	No	No	mini-batch
Range Loss [196]	Yes	Yes	No	No	mini-batch
L-Softmax Loss [90]	No	Yes	No	No	mini-batch
A-Softmax Loss [89]	No	Yes	Yes	No	mini-batch
AM-Softmax Loss [160]	No	Yes	Yes	Yes	mini-batch
ArcFace Loss [29]	No	Yes	Yes	Yes	mini-batch
GICO Loss*	Yes	Yes	Yes	Yes	global info

[1] WN: weight normalisation. FN: feature normalisation.

After reviewing the recent loss functions used in deep face recognition, we propose GICO Loss to combine the advantages of existing loss functions with some important new properties. Firstly, we apply L2 weight normalisation by fixing $b_j = 0$ and $||W_j|| = 1$. We also apply L2 normalisation on the feature vector f_i and re-scale $\|f_i\|$ to s. Similar

to Centre Loss, GICO Loss is used in conjunction with AM-Softmax Loss. Here we do not adopt Softmax Loss like Centre Loss, because AM-Softmax Loss shows slightly better performance than Softmax Loss. The total loss is thus:

$$\mathcal{L} = \mathcal{L}_{AM} + \lambda \mathcal{L}_G \tag{5.11}$$

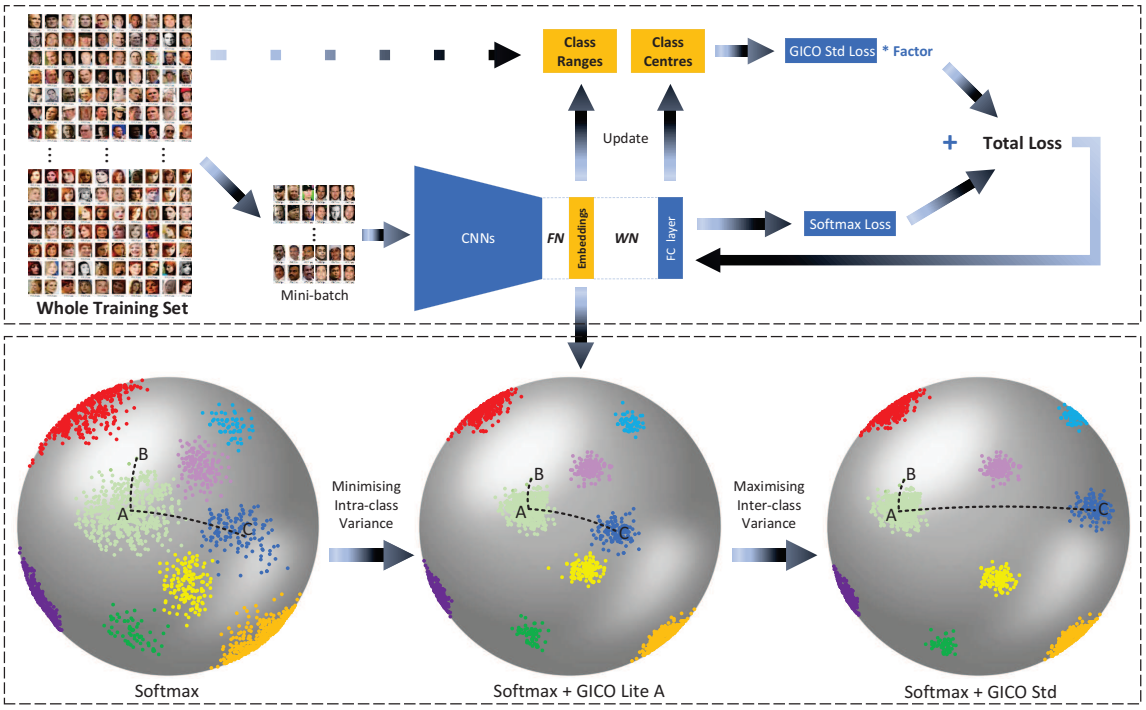


Figure 5.3: An overview of the proposed GicoFace framework. *FN* and *WN* represent *feature normalisation* and *weight normalisation*, respectively. A and C are the class centres of the corresponding classes. AB represents the class range and AC represents the distance between two class centres.

In designing the GICO Loss, two aspects are considered: minimising the intra-class variance and maximising the inter-class variance. These two aspects correspond to two "lite" versions of GICO Loss, respectively. Finally, we construct a standard version of GICO Loss, which is the combination of the two lite versions.

5.3.2 GICO Lite A

To minimise the intra-class variance, we propose a "lite" version of GICO Loss (GICO Lite A), which is formulated as below:

$$\begin{aligned} \mathcal{L}_{G_A} &= \frac{P}{\sum_{j=1}^{P} \frac{R(j)+1}{2}} \\ R(j) &= cos(c_j, e_j) \end{aligned} \tag{5.12}$$

where P is the number of classes in the whole training set, c_j is the centre of class j, and e_j denotes the edge of class j (i.e. the farthest sample to the centre of class j). $R(j)$ represents the cosine range of class j, namely the cosine similarity between the class centre and the edge of class j. During the training, the deep features are changing after each mini-batch, which also leads to changes in c_j and e_j. Ideally, c_j and e_j should be calculated by traversing the entire training set and should be updated after each mini-batch. However, this would require massive computing power that is completely impractical for the existing hardware. Commonly, a deep neural network is trained by iteratively updating the network parameters based on the feedback information from each mini-batch. This enables the network to have a stronger generation ability. Also, it is a practical solution within two constraints: the computing power and the memory size of GPU, TPU or other similar processing units. Without the computing power constraint, some accurate information can be calculated from the entire training set for training. Without the memory size constraint, the entire training set can be put into the memory and provides some global information to the losses, as some losses need precise global information, like the class centre and the class edge in GICO Loss and other losses.

Here we overcome these two constraints to use the entire training set as the source of feedback information by two approximate solutions. From Eq (5.3), we can see that the key optimisation object of the AM-Softmax Loss is actually minimising θ_{y_i} while maximising θ_j. θ_{y_i} is the angle between W_{y_i} and f_i. θ_j is the angle between W_j and f_i where $j \neq y_i$. In

other words, AM-Softmax Loss tries to reduce the distances between W_j and the sample features in the jth class ($j = 1, 2, ..., P$). As the training goes on, W_j is automatically optimised to the centre of class j ($j = 1, 2, ..., P$)[2], because this leads to the minimum distance sum between W_j and the sample features in the jth class. Therefore, we can simply use W_j as the substitution of c_j, which does not require any additional computing power. For e_j and $R(j)$, we propose ***a learning algorithm*** to recursively update the range of each class rather than directly calculating $R(j)$ with c_j and e_j. $R(j)$ is initialised to 1. Then we update $R(j)$ using the following iterations:

$$R(j)^{t+1} = R(j)^t + \sum_{i=1}^{N} \phi(y_i, j) \cdot \Delta R_i, j = 1, 2, ..., P. \tag{5.13}$$

$$\Delta R_i = \begin{cases} cos(W_{y_i}, f_i) - R(y_i)^t, & R(y_i)^t > cos(W_{y_i}, f_i) \\ \beta \cdot (cos(W_{y_i}, f_i) - R(y_i)^t), & R(y_i)^t \leq cos(W_{y_i}, f_i) \end{cases} \tag{5.14}$$

where $\phi(y_i, j) = 1$ when $y_i = j$, otherwise $\phi(y_i, j) = 0$. β is the shrink rate which is used to adjust the shrinking speed of the learned class range. The basic idea of the learning algorithm addresses two cases: (a). if the cosine similarity between the input sample and its corresponding class centre is smaller than the recorded class range, replace the class range directly with their cosine similarity; (b). if the cosine similarity between the input sample and its corresponding class centre is larger than the recorded class range, let the class range shrink by scaling their cosine similarities with β. Case (a) uses the newfound farthest sample to update the learned class range. However, as the training proceeds, the real class range will become smaller and smaller. So case (b) helps the learned class range to shrink to the real value.

[2]The truth that W_j will finally converge to the centre of class j is also described in Fig. 6 of [161]

5.3.3 GICO Lite B

To maximise the inter-class variance, we also propose another "lite" version of GICO Loss (GICO Lite B):

$$\begin{aligned} \mathcal{L}_{G_B} &= \frac{\sum_{Top}(A, K)}{K} \\ A &= \{\frac{cos(W_a, W_b) + 1}{2} : a, b = 1, 2, 3, ..., P; a > b\} \end{aligned} \tag{5.15}$$

where A is a set and $\sum_{Top}(A, K)$ denotes the sum of the K largest elements in A. The purpose of GICO Lite B is to find K pairs of nearest class centres in the entire training set and to calculate the sum of their distances. Compared with the non-adjacent class centres, the corresponding classes of the adjacent centres have a high probability of having small margins or having overlaps. If all adjacent classes have proper margins, the non-adjacent classes would have larger margins. Therefore, it is not necessary to take all centre pairs into account. The most effective way is to optimise the distances of all the adjacent centres. To judge whether a pair of class centres are adjacent, all the other class centres need to be checked whether they are on the shortest path between this pair of class centres. Therefore, it requires a recursive algorithm to do the operations in $O(P^3)$ time to find all adjacent centre pairs on the hypersphere, which is extremely time-consuming. Here we adopt a conservative and simple strategy, namely to set the value of K to P where P is the number of classes. Because the minimum number of adjacent centre pairs is P, which happens when each class centre has only two adjacent class centres and all the class centres align in a circle on the surface of the hypersphere.

5.3.4 GICO Std and Discussion

Finally, to achieve the best performance, we integrate GICO Lite A and GICO Lite B to create the standard version of GICO Loss (GICO Std):

$$\mathcal{L}_{G_{std}} = L_{G_A} * L_{G_B} = \frac{P * \sum_{Top}(A, K)}{K * \sum_{j=1}^{P} \frac{R(j)+1}{2}} \tag{5.16}$$

GICO Std is more complex, but it simultaneously optimises the intra-class variance and inter-class variance, which leads to better performance. Combining CNNs with the proposed GICO Losses, we build a high-performance deep model named GicoFace, as shown in Fig. 5.3.

- **Why is global information introduced?** Optimising the global sample distribution is the intrinsic purpose of training. The training based on mini-batches is a trade-off because of the hardware constraints. Therefore, it is reasonable to consider using global information if hardware constraints can be overcome. The introduction of global information into training leads to a better sense of the ”big picture” rather than focussing on only each mini-batch.

- **Why is the information from each mini-batch still being used?** GICO Loss is applied along with AM-Softmax Loss, so the information from each mini-batch is also used in training. Since the feedback information of a mini-batch is directly from the real samples, the network parameters can be updated in time to fit these samples, ensuring an acceptable training speed. Due to the hardware constraints, the global information introduced by GICO Loss is obtained by learning. This learning process decreases the interaction speed between the network parameters and GICO Loss. As a result, using GICO Loss alone will lead to a slow training process. Training speed is slowest at the early stage of the training as the learned class ranges and the class centres are not steady at the initial stage. Hence, it is necessary to use both global information and information from each mini-batch.

5.3.5 Experiments

5.3.5.1 Experiment Settings

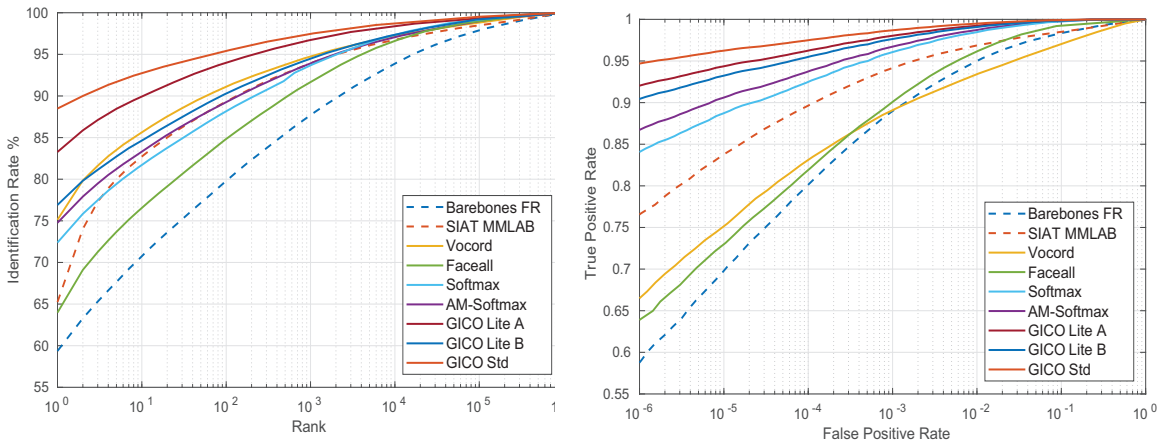
Our deep face models are implemented by Tensorflow[3] with Inception-ResNet-v1 [144] as the trunk network. We combine Inception-ResNet-v1 with different loss functions, resulting in 5 different combinations: (1). ResNet+Softmax, (2). ResNet+AM-Softmax, (3). ResNet+GICO Lite A, (4). ResNet+GICO Lite B, and (5). ResNet+GICO Std.

In all experiments, we set 320 as the epoch size, 120 as the batch size, 5e-4 as the weight decay, 0.4 as the keep probability of the fully-connected layer, 512 as the embedding size and 0.01 as the shrink rate. We manually optimise the hyperparameter λ. Since performance is not very sensitive to the value of λ, we just try multiple different values, and choose the value that leads to the lowest total loss. The initial learning rate is set to 0.05 and is reduced by a factor of 10 every 100,000 iterations.

VGGFace2 [21] is the training set in all of our experiments. To guarantee the reliability of the results, we removed the people who might overlap with the testing sets from VGGFace2. We did not do any noise removal as VGGFace2 is a very clean dataset. Finally, the preprocessed training set contains 3.05 million face images. For testing, we use five public benchmark datasets: LFW [58], SLLFW [195], YTF [178], MegaFace [68] and FaceScrub [105] datasets. For image preprocessing, we applied the same pipeline of processes on every raw image in the training set and the testing sets. Firstly MTCNN [194] is employed for face detection. MTCNN occasionally fails to detect the face. If this occurs for a training image, the image is simply abandoned. If it occurs for a testing image, we use the provided official landmarks or bounding boxes instead. All the face images are cropped to the size of 160*160. To enhance the randomness of the training data, every training image is randomly horizontally flipped before being input to the network. The final features of a testing image are generated by concatenating the features of the original image and the features of its horizontally flipped counterpart so as to improve the recognition accuracy.

[3] https://www.tensorflow.org/

5.3.5.2 MegaFace Challenge 1 on FaceScrub



(a) (b)

Figure 5.4: (a) The CMC curves of different methods with 1 million distractors on MegaFace Set 1. (b) The ROC curves of different methods with 1 million distractors on MegaFace Set 1.

In this section, we evaluate the performance of the proposed GICO Loss on the MegaFace dataset [68] and the FaceScrub dataset [105]. We follow the experimental protocol of MegaFace Challenge 1. This protocol allows participants to train their models on any training set but requires the participants to do identification and verification testing on the FaceScrub dataset and use the MegaFace dataset to set 1 million face images as the distractors. The evaluation is conducted with the officially provided code [68]. Fig. 5.4(a) and Fig. 5.4(b) report the CMC curves and the ROC curves of different methods with 1M distractors on MegaFace Set 1, respectively. The results of the benchmark methods (including Barebones FR, SIAT MMLAB, Vocord and Faceall) are generated with the evaluation code and features provided by MegaFace team[4]. From Fig. 5.4(a), we observe that all three versions of GICO Loss outperform Softmax, AM-Softmax and other benchmark methods on the Rank1 identification rate. GICO Std shows better performance than GICO Lite B and GICO Lite A, while GICO Lite A performs better than GICO Lite B. Fig. 5.4(b) shows the

[4]The features of the benchmark methods provided by the MegaFace team: (http://megaface.cs.washington.edu/participate/challenge.html)

Table 5.4: Verification performance of state-of-the-art methods on LFW and YTF datasets.

Methods	Source	Images	LFW(%)	YTF(%)
Range Loss [196]	ICCV17'	1.5M	99.52	93.7
DeepID2+ [142]	CVPR15'		99.47	93.2
Deep Face [148]	CVPR14'	4M	97.35	91.4
Fusion [149]	CVPR15'	500M	98.37	
FaceNet [131]	ICCV15'	200M	99.63	95.1
Centre Loss [173]	ECCV16'	0.7M	99.28	94.9
Multibatch [147]	NIPS16'	2.6M	98.20	
Aug [99]	ECCV16'	0.5M	98.06	
L-Softmax [90]	ICML16'	0.5M	98.71	
A-Softmax [89]	CVPR17'	0.5M	99.42	95.0
Softmax	N/A	3.05M	99.50	95.22
AM-Softmax	N/A	3.05M	99.57	95.62
GICO Lite A	N/A	3.05M	99.60	95.70
GICO Lite B	N/A	3.05M	99.62	95.78
GICO Std*	N/A	3.05M	99.63	95.82

verification performance, where all three versions of GICO Loss significantly outperform the other methods. GICO Std still shows better performance than GICO Lite B and GICO Lite A. These results on FaceScrub dataset demonstrate the effectiveness of the proposed GICO Loss.

5.3.5.3 Results on LFW, YTF and SLLFW

In this section, we compare the proposed methods with the state-of-the-art methods on LFW, YTF and SLLFW. LFW and YTF stipulate multiple standard experimental protocols. In our experiments, we follow the standard experimental protocol of "unrestricted with labelled outside data" [57]. Following this protocol and the given pair list, we do the verification test on 6,000 face pairs in LFW including 3,000 positive pairs (same identity) and 3,000 negative pairs (distinct identities), and we do the verification test on the given 5,000 video pairs in YTF including 2,500 positive pairs and 2,500 negative pairs.

Under the same experimental protocol, we compare the results of the proposed method and the state-of-the-art methods on LFW and YTF, as shown in Table 5.4. The results of the benchmark methods shown in the upper part of the table are cited from their original

Table 5.5: Verification performance of different methods on SLLFW.

Method	Source	Images	LFW(%)	SLLFW(%)
Deep Face [148]	CVPR14'	0.5M	92.87	78.78
DeepID2 [139]	NIPS14'	0.2M	95.00	78.25
VGG Face [114]	BMVC15'	2.6M	96.70	85.78
DCMN [34]	PR[J]17'	0.5M	98.03	91.00
Noisy Softmax [23]	CVPR17'	0.5M	99.18	94.50
Softmax	N/A	3.05M	99.50	96.17
AM-Softmax	N/A	3.05M	99.57	98.02
GICO Lite A	N/A	3.05M	99.60	98.15
GICO Lite B	N/A	3.05M	99.62	98.13
GICO Std*	N/A	3.05M	99.63	98.17

papers. From Table 5.4, we observe the following. GICO Std shows higher verification accuracy on LFW than Softmax, AM-Softmax, GICO Lite A and GICO Lite B. GICO Std ties with FaceNet for first place on LFW. However, FaceNet uses 200 million images for training, while GICO Std uses only 3.05 million images. GICO Std also beats the other benchmark methods on LFW, most of which are published in leading computer vision conferences. On YTF dataset the proposed GICO Loss still has better performance than most of the benchmark methods, which demonstrates the state-of-the-art performance of the GICO Loss.

LFW is a popular face dataset, but more and more methods are gradually touching its theoretical upper limit[5]. Consequently, it becomes more and more difficult to differentiate different methods on LFW. To confirm the performance of the proposed methods, we conducted an additional experiment on SLLFW [195]. SLLFW uses the same positive pairs as LFW for testing, but in SLLFW, 3000 similar-looking face pairs are deliberately selected from LFW by human crowdsourcing to replace the random negative pairs in LFW. SLLFW adds more challenges to the testing, causing the accuracy of the same state-of-the-art methods to drop by 10-20%.

Table 5.5 shows the verification accuracy of different methods on SLLFW. The results

[5]There are six mismatched pairs on LFW which are incorrectly labelled as matched. So the upper limit accuracy on LFW is (6000-6)/6000=99.90%.

of some benchmark methods are shown in the top half of the table. These results are publicly accessible and provided by the SLLFW team[34]. As can be seen from Table 5.5, GICO Loss achieves considerably better performance than other methods on SLLFW. In the top half of the table, the accuracy of the benchmark methods drops by between 16.75% and 4.68% from LFW to SLLFW. By comparison, the accuracy of GICO Loss drops by between 1.45% and 1.49%. The results on SLLFW further confirm the performance of the proposed methods.

5.4 Summary

This chapter focuses on Cosine similarity-based loss and presents two Cosine similarity-based losses – Precise Adjacent Margin Loss (PAM Loss) and Global Information-based Cosine Optimal Loss (GICO Loss). In this chapter, we proposed the PAM Loss to precisely optimise the real edge-to-edge margin. To make PAM Loss possible, we also proposed a learning algorithm to obtain the range of each class. Extensive experiments are conducted on LFW [58], YTF [178], MegaFace [68] and FaceScrub [105] datasets. Results demonstrate the effectiveness of the proposed methods and confirm the state-of-the-art performance of PAM Loss. As PAM Loss considers only optimising the inter-class distance, we propose a more complete loss function GICO Loss in Section 5.3 to simultaneously optimise both inter-class and intra-class distance. GICO Loss integrates the advantages of the best loss functions proposed in recent years in face recognition. To make GICO Loss possible, we propose a novel algorithm to learn the cosine similarity between the class centre and the class edge. Extensive experiments are conducted on LFW, SLLFW, YTF, MegaFace and FaceScrub datasets. Results demonstrate the effectiveness of the proposed GICO Loss and show that it achieved state-of-the-art performance.

CHAPTER VI

A Complete Face Search Framework for Image and Video Retrieval

6.1 Introduction

A face search framework is the combination of a face recognition framework and a search engine. A face search framework can quickly and accurately find similar-looking faces to a given face from a large collection of faces, and return a collection of similar faces. Face search has a wide range of applications such as intelligent album, face-based gate control, and smart surveillance. Besides these, automatic face tagging is especially high in potential among various applications. Social media, like Facebook, Twitter, Instagram, and Youtube, generate billions of photos and video frames every day. Given a photo or a video clip, a face search framework can provide effective tagging suggestions which can improve the efficiency and reliability of data access.

In this chapter, we describe the design and implementation of our complete face search framework. We define a complete face search framework as a framework that can retrieve both image and video, and has the front-end and the back-end, where the front-end should include the search interface and other functional interfaces, while the back-end should include the following modules: face detection, face alignment, face cropping, model training, feature extraction, database management, indexing and retrieval.

6.2 Motivation

Much work [75, 159, 138, 103] has been done for developing a face search framework. However, they implement only some of the aforementioned parts, and so can not be called a complete face search framework according to our definition. Moreover, they did not employ the current state-of-the-art techniques in each module. Commercial face search frameworks include Betaface [1], PicTriev [5] and PimEyes [6]. When a face image is uploaded to Betaface, Betaface retrieves their celebrity database or wikipedia database to return the identity of the uploaded face. PicTriev can find look-alike celebrities on the web with the face image and provide predictions regarding three face attributes (i.e. masculine, feminine and age). Nevertheless, PicTriev has high requirements on the resolution and quality of the face image, and accepts only jpg(jpeg) image format with image size less than 200K bytes. PimEyes can find faces appearing on the web that are similar to the provided face image, and can provide the website link where the similar face appears on. However, Betaface, PicTriev and PimEyes can only search for faces in images, not videos, and none of them publishes their implementation details or reports their performance.

6.3 The Framework

This section introduces the design of the proposed framework from its structure to its usage. Fig. 6.1 illustrates the proposed face search framework. From Fig. 6.1, we can see that the proposed framework has three major parts — (a) preprocessing, (b) feature learning and extraction, and (c) indexing and retrieval. Each major part has some minor modules, where preprocessing includes face detection, face alignment and face cropping; feature learning and extraction consists of Convolutional Neural Networks (CNNs), loss function and a trained network model; and indexing and retrieval incorporate indexing the feature database and retrieving the feature database with the features of the query image.

As Fig. 6.1 shows, the proposed framework has the following four function routes: (a)

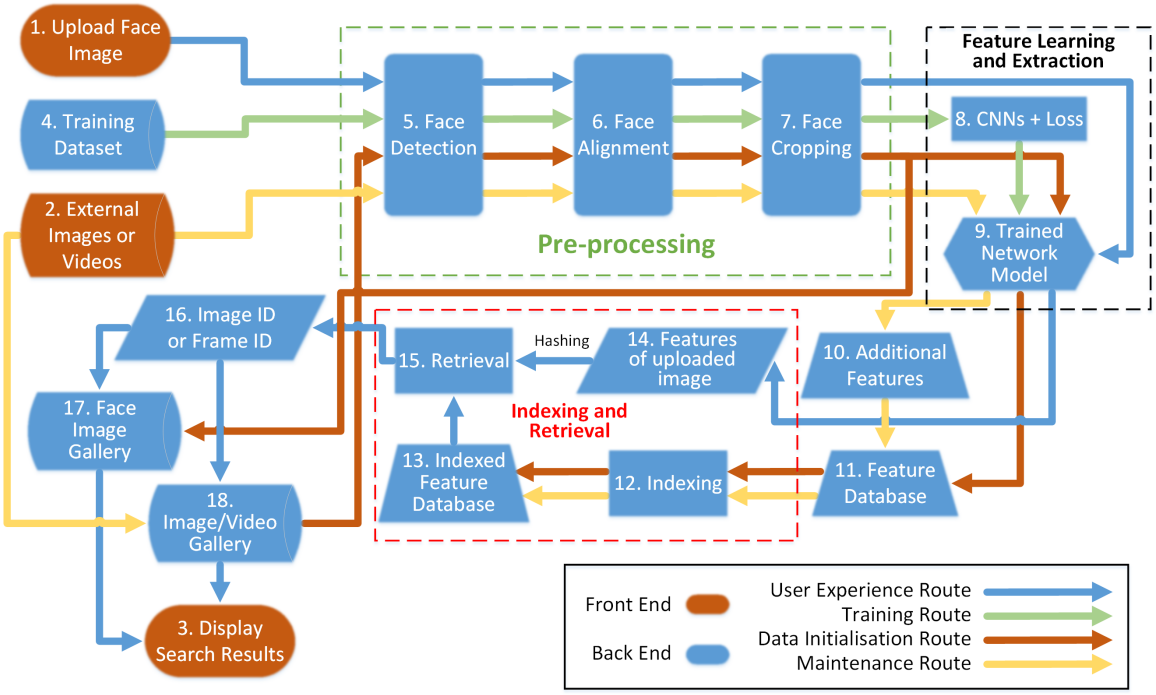


Figure 6.1: Flowchart of the proposed face search framework.

user experience route, (b) training route, (c) data initialisation route, and (d) maintenance route. Before the user starts to search for a face with the framework, the administrator needs to take the training route to obtain a proper network model, and take the data initialisation route to execute a series of processes on the image and video gallery. When going through the training route, the framework firstly preprocesses the training data, then inputs the preprocessed data to the CNNs along with an appropriate loss function. After training, a trained network model can be obtained for feature extraction in the future. When going through the data initialisation route, the image/video gallery is firstly preprocessed with the same techniques, then their features are extracted with the trained network model and are gathered to build a complete feature database. Next, the feature database is indexed by hash coding to get an indexed feature database for future use.

With the above preparation, users can experience the face search framework by entering from the project homepage as shown in Fig. 6.2. To search for a face, users just need to upload an image of the face and choose the type of search result (image or video). The

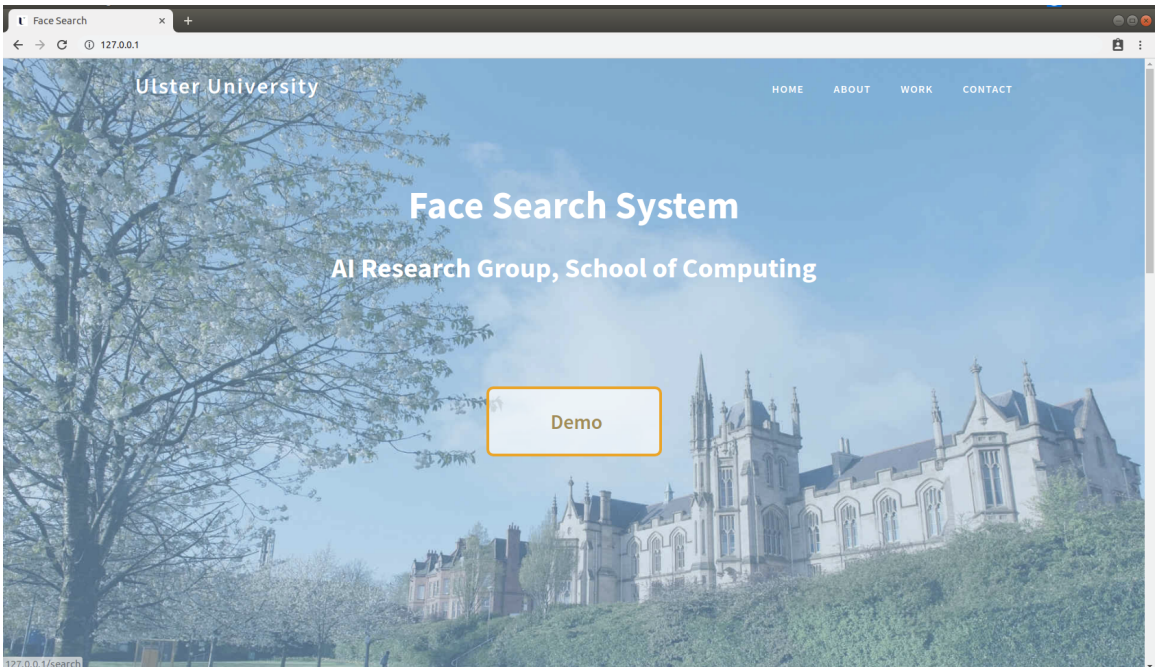


Figure 6.2: Cover page of the proposed face search framework.

framework will automatically process the uploaded image, retrieve all the images or videos in the gallery according to the blue route in Fig. 6.1, and finally list the most similar results. The images and videos are listed along with their detailed information, where the images can be downloaded and the video can be watched instantly from the frame with the highest similarity. Examples of face search on images and videos are shown in Fig. 6.3 and Fig. 6.4, respectively.

An administrator may want to manage the gallery by adding new images or videos, or deleting existing ones. To achieve this goal, the administrator just needs to run a one-key update program which executes the modules according to the maintenance route shown in Fig. 6.1. As Fig. 6.5 shows, users can also add their preferred videos to the library by sending their applications to maintainers. To simplify the flowchart, the maintenance route shown in Fig. 6.1 shows only the process for adding images or videos. To delete an image or a video, the whole process includes: (1) delete the original image or video from module 18, (2) delete corresponding preprocessed face images from module 17, (3)

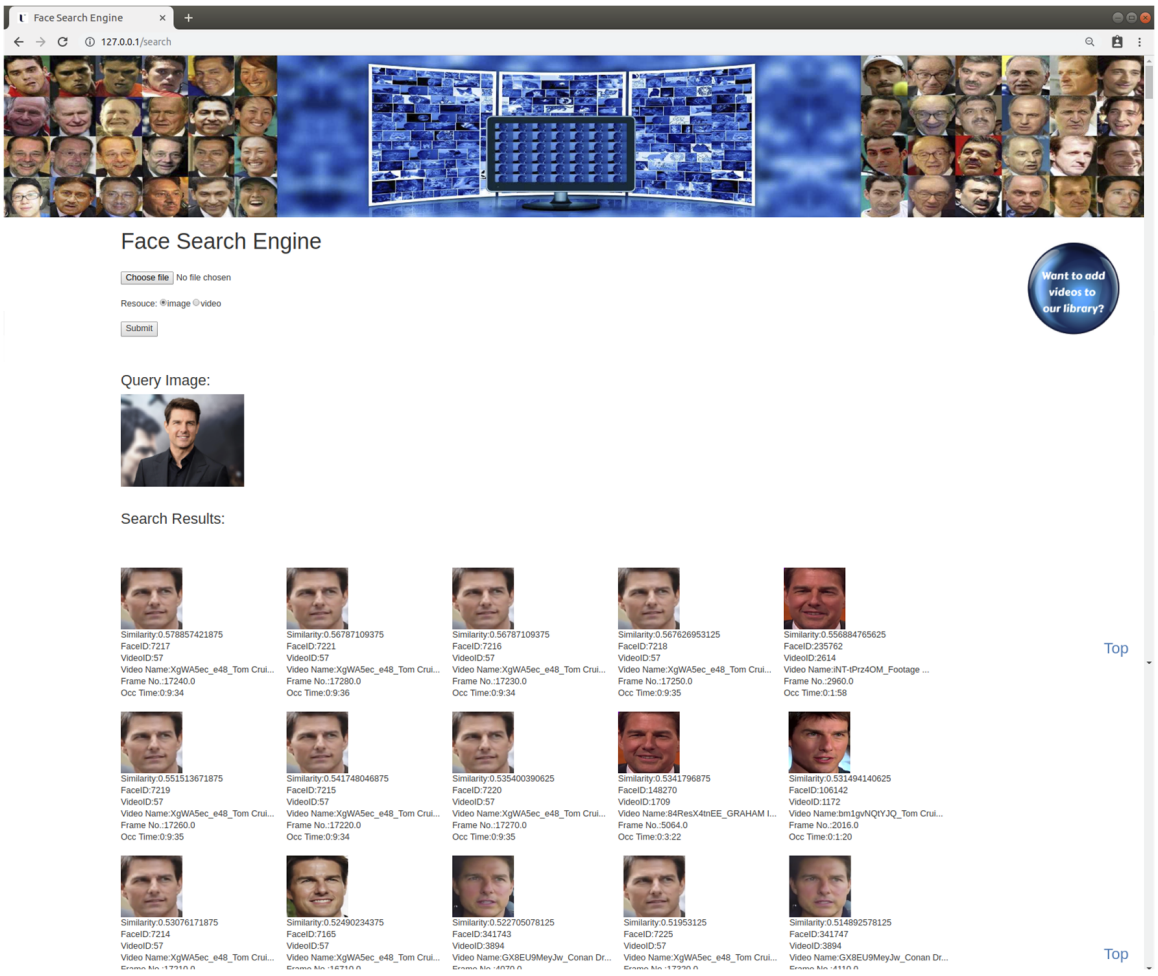


Figure 6.3: Example of face search on images.

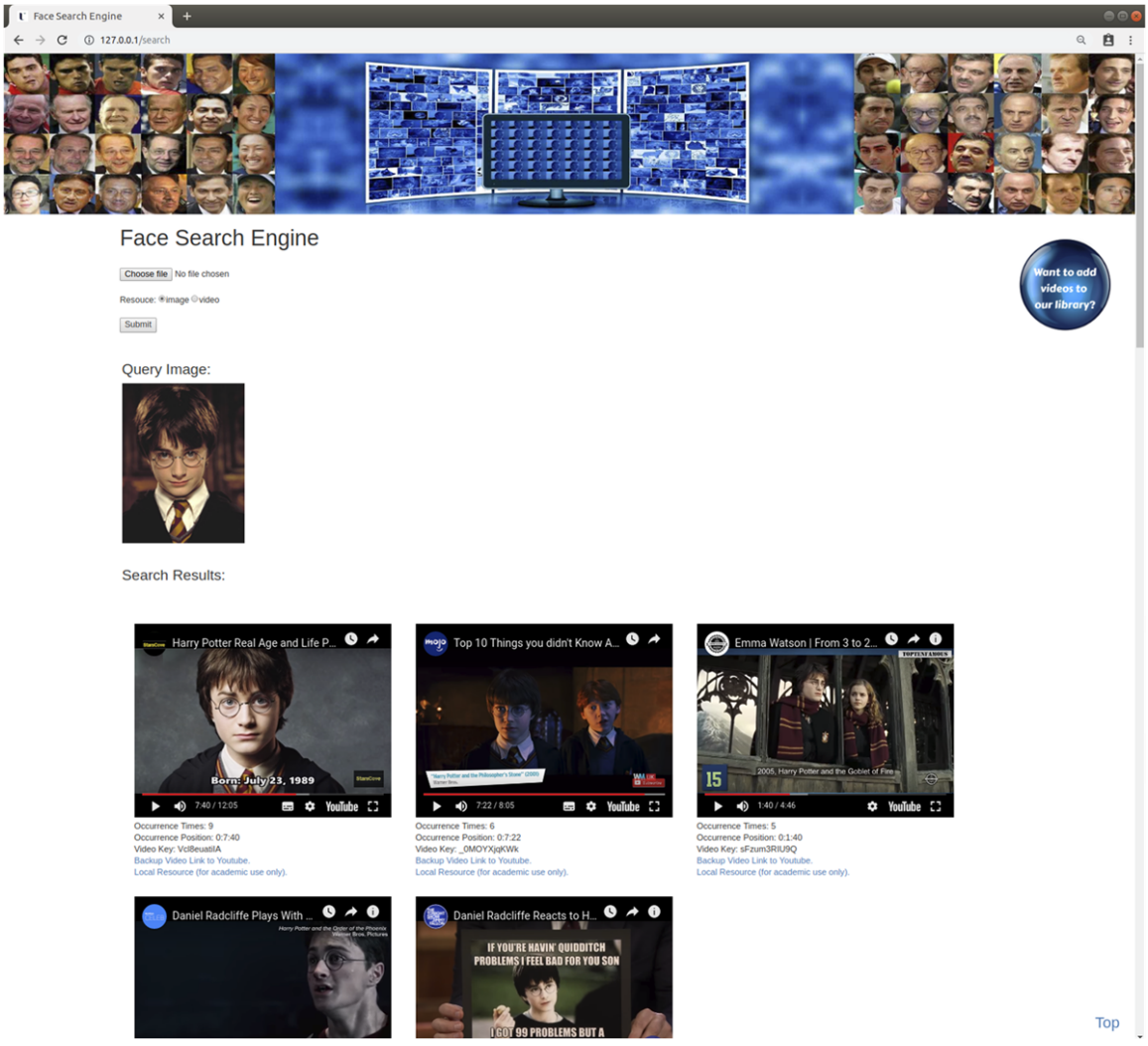


Figure 6.4: Example of face search in videos.

delete the corresponding features from module 11 and (4) delete the corresponding indexed features from module 13.

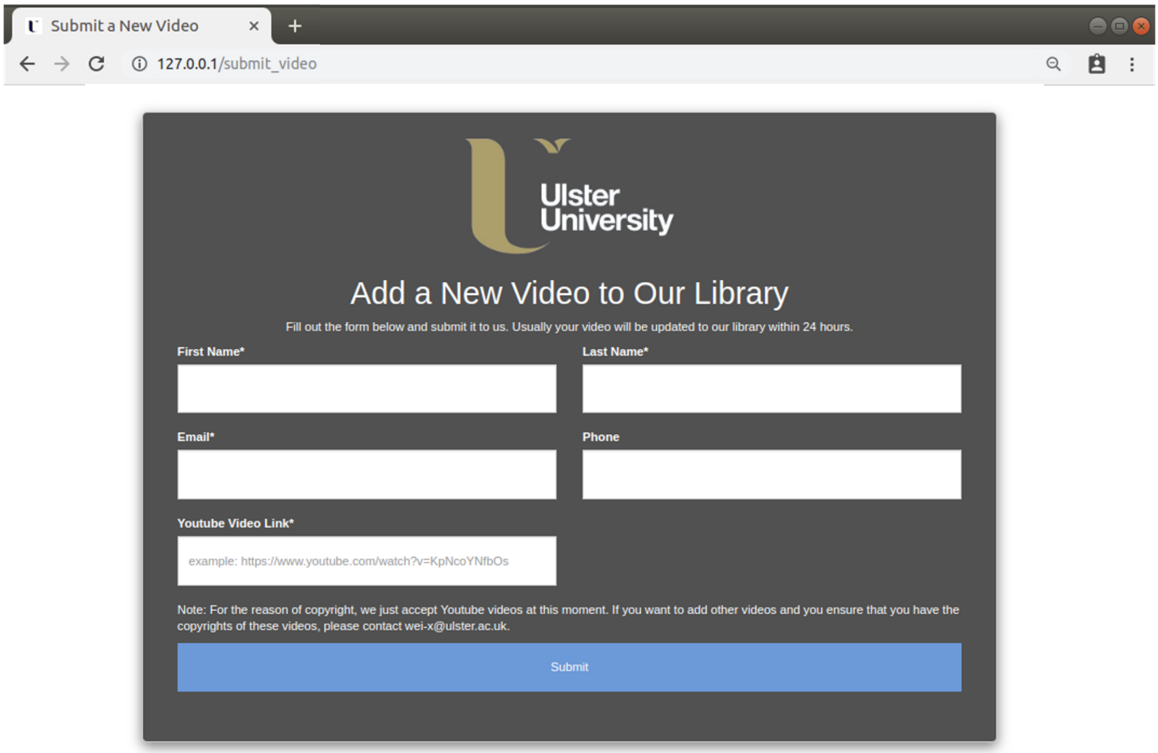


Figure 6.5: The web page for adding more videos.

6.4 A Demonstrator

In this section, we outline the implementation details of the proposed framework from module 1 to module 18 as illustrated in Fig. 6.1. The front-end consists of module 1 to module 3, which are implemented by HTML5, CSS and JavaScript. To avoid copyright infringement, the video search results are displayed by embedding the original Youtube videos as shown in Fig. 6.4. An example of our embed code is as below:

```
<iframe
    width=420" height="315"
    src="https://www.youtube.com/embed/{{s[5]}}?start={{s[3]}}"
```

```
  frameborder="0" allowfullscreen>
</iframe>
```

The back-end consists of module 4 to module 18. These modules are mainly implemented by Python, Tensorflow and Flask. In module 4, VGGFace2 [21] is used as our training set. We filtered the VGGFace2 dataset in the same way as previous chapters. The resulting training set consists of 3.05M facial images from 8K identities. From module 5 to module 7, we apply the Multi-Task Cascaded Convolutional Neural Network (MTCNN) [194] for face detection and utilise the landmarks detected by MTCNN for face alignment and face cropping. We choose MTCNN because it is very stable and achieved the state-of-the-art results on a range of benchmark datasets. However, MTCNN still has a small probability of failing on detection. If it fails on a training image, we just remove that image from the training set. If it fails on an uploaded face image, a prompt box will pop up and advise the user to upload another image. Users should make sure that their uploaded image contains at least one face.

In module 8, we implemented Inception-ResNet-v1 [144] and Inception-ResNet-v2 [145] as the network architectures. These two architectures are very competitive in face recognition even compared with the state-of-the-art architectures. Following our research on loss functions in previous chapters, we have implemented a wide range of loss functions for selection in module 8. By combining different loss functions with different architectures, we can train different models. Here we set Inception-ResNet-v1 + PAM Loss v1 as our default combination as it shows the best performance in experiments. With the trained network model in module 9, the features in module 10, 11 and 14 can be extracted from the preprocessed face images. The features extracted from the gallery are then gathered together to build a continuous binary file (i.e. the feature database in module 11) for fast reading and writing.

To accelerate the retrieval speed, we implemented two hash coding methods — Iterative Quantization (ITQ) [46] and Locality Sensitive Hashing (LSH) [11] for indexing in module

12, where ITQ is set as the default option. Using ITQ or LSH, a hash projection matrix can be computed based on the feature database, with which the feature database can be converted to an indexed feature database in module 13, and the features of a single image in module 14 can also be converted into hash codes for retrieval. In module 15, the hash codes of the query image are compared with the hash codes of the images or video frames in the database one by one. We calculate the Euclidean distance of the hash codes of two faces, and use the reciprocal of the Euclidean distance as the similarity of two faces. Sorting the face images by similarity, we get a retrieval ranking of image ID or frame ID.

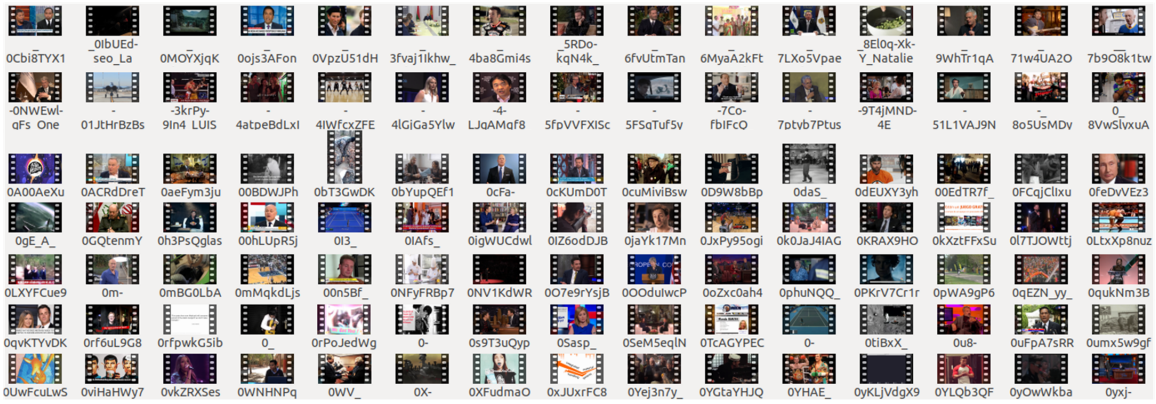


Figure 6.6: Example of the videos in the gallery.

Figure 6.7: Example of the preprocessed faces in the proposed framework.

Module 18 is an image/video gallery that can store all kinds of original images or

videos. At present, we put 5,167 videos into this gallery; examples are shown in Fig. 6.6. To collect these videos, we build a name list containing 134 celebrities and use their names as the keywords to search on YouTube. From YouTube, an average of 38.5 videos is collected for each celebrity. For each video, we extract its frames at an interval of 0.5 seconds and use modules 5 to 7 to process all these frames. The resulting and preprocessed face images are then stored in a face image gallery in module 17. Examples of the preprocessed faces are shown in Fig. 6.7. For each face image in the face image gallery, we save its relevant information into a table, which includes its unique id, its file path, its frame number in the video, its occurrence time in the video, the id and the name of the video to which the frame belongs. Based on the above preparation, we can display the search results. If searching for images, image ID and frame ID will be obtained (module 16) and then be used to query the aforementioned table to fetch the face images and their corresponding information for display. If searching for videos, the resulting videos are sorted according to the proposed Eq. 6.1.

$$R_i = \sum_{j=1}^{N_i} \frac{min(D(F_{ij}, P) - T, 0)}{D(F_{ij}, P) - T} \tag{6.1}$$

where R_i is the relevance between the ith video and the query face image P, N_i is the total number of the extracted frames from the ith video, F_{ij} is the jth frame from the ith video, $D(\cdot, \cdot)$ represents the Euclidean distance of two images, T is the threshold for excluding some dissimilar frames, and $min(\cdot, \cdot)$ represents the minimum of two numbers. Eq. 6.1 computes the relevance by counting the number of frames that have a distance smaller than the specified threshold T. Moreover, the videos that have a relevance value smaller than ξ are filtered out, which can exclude some false accepted videos. Finally, the resulting videos are displayed by relevance from high to low.

6.5 Functional Evaluation

The proposed framework has three main features: a) high retrieval accuracy, b) low requirement on uploaded images and c) support for face search in videos. This section describes the details of these three features and evaluates these three features. Much research [75, 159, 138, 103] has been done for developing a face search framework. However, we cannot directly compare our framework with these frameworks. The reason is two-fold: they did not publish their codes or datasets, making us unable to conduct evaluation under the same environment, and they are not complete face search frameworks according to our definition, and they have different design targets and features. Therefore, it is also unfair and nearly impossible to cite their results for comparison. Commercial face search frameworks include Betaface [1], PicTriev [5] and PimEyes [6], and some typical cases are used for comparing Betaface, PicTriev and PimEyes with the proposed framework.

6.5.1 High Retrieval Accuracy

We randomly uploaded 30 celebrity face images as the query images to test the performance of Betaface, PicTriev and PimEyes and the proposed framework. Their average search time is 4.9s, 8.2s, 7.3s and 5.9s, respectively. However, we found the search time was largely influenced by internet speed, so the search time may change under different conditions. The search results show that 99.95% of the top-100 images found by our framework are correct. PimEyes has an accuracy of 95.82% in terms of the top-100 resulting images. Different from PimEyes and the proposed framework, Betaface and PicTriev return the results in the form of individuals, so we report their top-1 results. Based on the top-1 results, Betaface and PicTriev correctly recognise 83.33% and 43.33% of the individuals, respectively. Three of the testing examples are shown in Fig. 6.8.

The high retrieval accuracy of the proposed framework is mainly due to its two major parts — (a) feature learning and extraction, and (b) indexing and retrieval. In feature learning and extraction, the network model is the core which determines the performance of this

Figure 6.8: Examples of image search results of different frameworks.

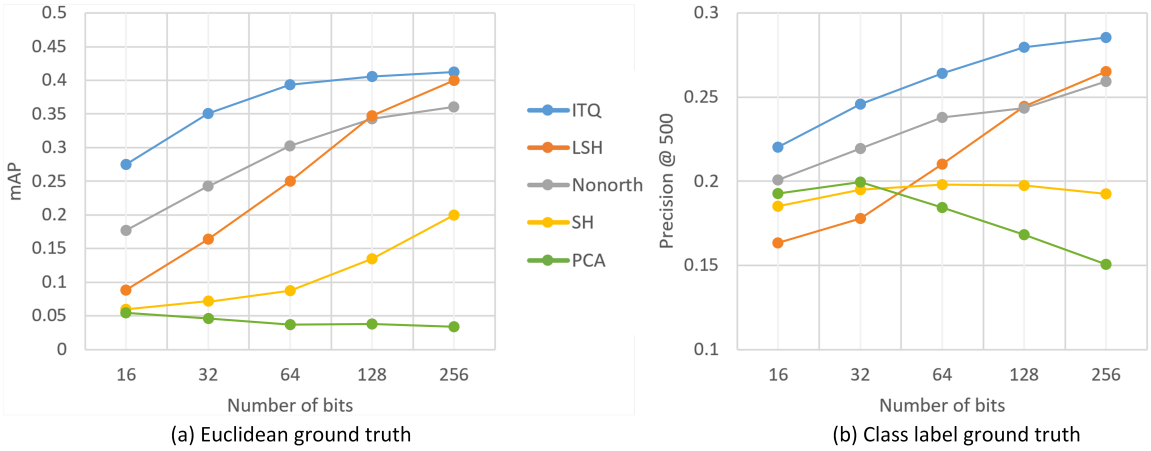
Table 6.1: Verification rates of state-of-the-art models on LFW and YTF datasets.

Methods	Source	Images	VR on LFW(%)	VR on YTF(%)
Range Loss [196]	ICCV17'	1.5M	99.52	93.7
Marginal Loss [30]	CVPR17'	4M	99.48	96.0
VGG Face [114]	BMVC15'	2.6M	98.95	97.3
Deep Face [148]	CVPR14'	4M	97.35	91.4
FaceNet [131]	ICCV15'	200M	99.63	95.1
Centre Loss [173]	ECCV16'	0.7M	99.28	94.9
Multibatch [147]	NIPS16'	2.6M	98.20	
Aug [99]	ECCV16'	0.5M	98.06	
SphereFace [89]	CVPR17'	0.5M	99.42	95.0
Contrastive CNN [49]	ECCV18'	0.5M	99.12	
OE-CNNs [168]	ECCV18'	1.7M	99.47	
MC Loss	N/A	3.05M	99.57	95.34
MML	N/A	3.05M	**99.63**	95.52
PAM Loss v1	N/A	3.05M	**99.63**	**96.14**
PAM Loss v2	N/A	3.05M	99.62	96.00
GICO Lite A	N/A	3.05M	99.60	95.70
GICO Lite B	N/A	3.05M	99.62	95.78
GICO Std	N/A	3.05M	**99.63**	95.82

part for face recognition. In indexing and retrieval, the key is how to recode the features to the hash codes. Next, we report on the performance of the proposed system for these two sub-tasks: 1) face recognition, and 2) indexing and retrieval.

Face Recognition: All the proposed deep models in previous chapters can be used in this framework. We compare these proposed models and the state-of-the-art models on LFW image dataset [58] and YTF video dataset [178]. The comparison results are shown in Table 6.1, where the upper part of the table shows the state-of-the-art models and the lower part of the table shows the proposed models. The results of the state-of-the-art models are cited from their papers. The implementation details of the proposed models have been presented in previous chapters. Table 6.1 shows that the proposed models outperform most of the existing models on LFW and YTF. The proposed models have a very similar performance on LFW, but PAM Loss v1 shows the best performance on both YTF and LFW dataset compared with other proposed models.

Indexing and Retrieval: Experiments are conducted on CIFAR dataset [73] to com-



(a) Euclidean ground truth

(b) Class label ground truth

Figure 6.9: Results of different indexing methods on CIFAR dataset.

pare the implemented ITQ [46] and LSH [11] with some other benchmark methods including Nonorth [164], (Spectral Hashing) SH [172] and (Principal Component Analysis) PCA [177]. ITQ and Nonorth use PCA as an intermediate dimensionality reduction step, while LSH and SH only rely on randomised data-independent linear projections. We follow two widely used experimental protocols. The first protocol takes the Euclidean distance-based neighbours as the ground truth to evaluate the performance of searching for the nearest neighbour. The mean distance to the 50th nearest neighbour is set as the threshold for separating positive and negative samples. The second protocol takes the class labels as the ground truth to evaluate different methods on semantic consistency. In the experiments, 1000 images are randomly selected from CIFAR dataset as the test queries. The rest of the images in the dataset are used for parameter learning and retrieval test. The test results are the average values over five random repetitions.

With the first protocol, we report the mean average precision (mAP) under different number of bits, as shown in Fig. 6.9(a). From Fig. 6.9(a), we can see that ITQ outperforms all the other methods at all code sizes. LSH shows poor performance with a small code size, but improves quickly with the increase of code size and almost reaches the level of ITQ at 256 bits. With the second protocol, we report the averaged precision of the top 500 ranked

images in Fig. 6.9(b). From 16 bits to 256 bits, ITQ still shows the best performance. LSH has the lowest precision at 16 bits, but its precision increases quickly with increase in code size. Using PCA alone, the precision drops rapidly with increase in code size.

6.5.2 Low Requirement on Uploaded Image

Compared with Betaface, PicTriev and PimEyes, the proposed framework has relatively low requirement on uploaded images. Our framework supports low-resolution images (as low as 120*120 pixels), supports all commonly used image formats, allows large pose variation, allows incomplete faces and has no limit on image size. In contrast, Betaface, PicTriev and PimEyes have a relatively low tolerance for the above situations. For example, PicTriev advises the user to upload photos of frontal faces, desirably with the gap between the eyes more than 80 pixels wide. In addition, PicTriev only accepts image files of jpg(jpeg) format with size below 200K bytes.

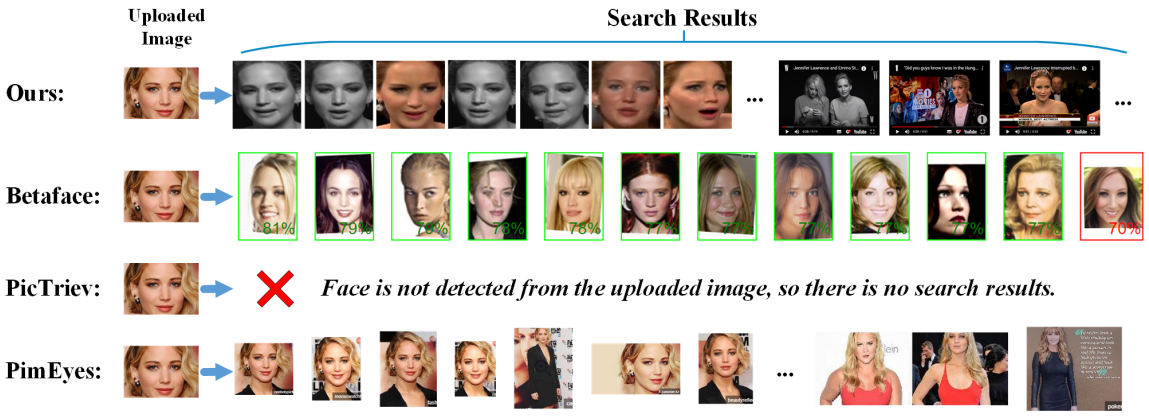


Figure 6.10: Search results of different face search framework on the same uploaded face image.

Fig. 6.10 shows an example of a search using different frameworks, where an image of an incomplete face is uploaded. From the results, we see that the proposed framework successfully found the image results and video results without any errors. Betaface completely failed to recognise the uploaded face. PicTriev failed to detect the face in the image.

PimEyes was more successful than Betaface and PicTriev, but had two errors.

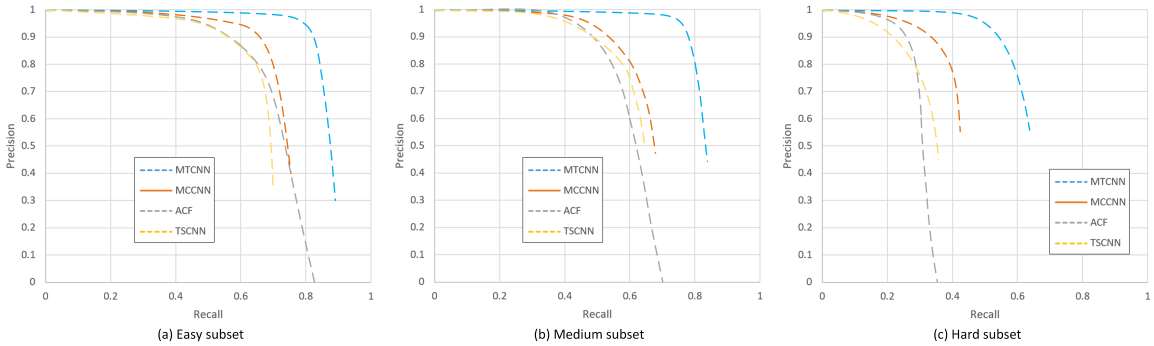


Figure 6.11: Results of face detection methods on three subsets of WIDER FACE.

To enable the proposed framework to reduce the requirement for image quality, a face detection algorithm must be robust enough. Otherwise, a face may not be able to be detected or the algorithm generates inaccurate bounding boxes. Inaccurate bounding boxes will lead to unreliable recognition results. In our framework, we implemented Multi-Task Cascaded Convolutional Neural Network (MTCNN) for face detection. Our MTCNN is trained in the same way as [194]. To evaluate its performance, we conduct experiments on three subsets of the WIDER FACE dataset [189]. WIDER FACE contains 32,203 images and 393,703 labelled faces with large intra-class variations, such as scale, pose and occlusion. Based on the detection rate of EdgeBox [200], WIDER FACE is divided into three subsets, namely 'Easy', 'Medium' and 'Hard' subsets. The implemented MTCNN is compared with multiple state-of-the-art methods including Multiscale Cascade CNN (MCCNN) [189], Aggregate Channel Features (ACF) [186] and Two-stage CNN (TSCNN)[189]. As shown in Fig. 6.11, the implemented MTCNN consistently outperforms all the compared methods by a large margin on all three subsets. It is also impressive that the implemented MTCNN still has a precision higher than 99% when the recall rate is 70% on the medium subset, however, the precisions of all the compared methods are less than 90% in this case, which indicates that the compared methods have lost their effectiveness.

6.5.3 Support Face Search in Videos

Different from the existing research works [75, 159, 138, 103] and the existing commercial face search frameworks [1, 5, 6], the proposed framework supports face search in videos and achieves a high retrieval accuracy. We randomly uploaded 30 celebrity face images as the query images, and the results show that 99.58% of the videos found are correct and 99.17% of the video locations are correct. Five of testing examples are shown in Fig. 6.12.

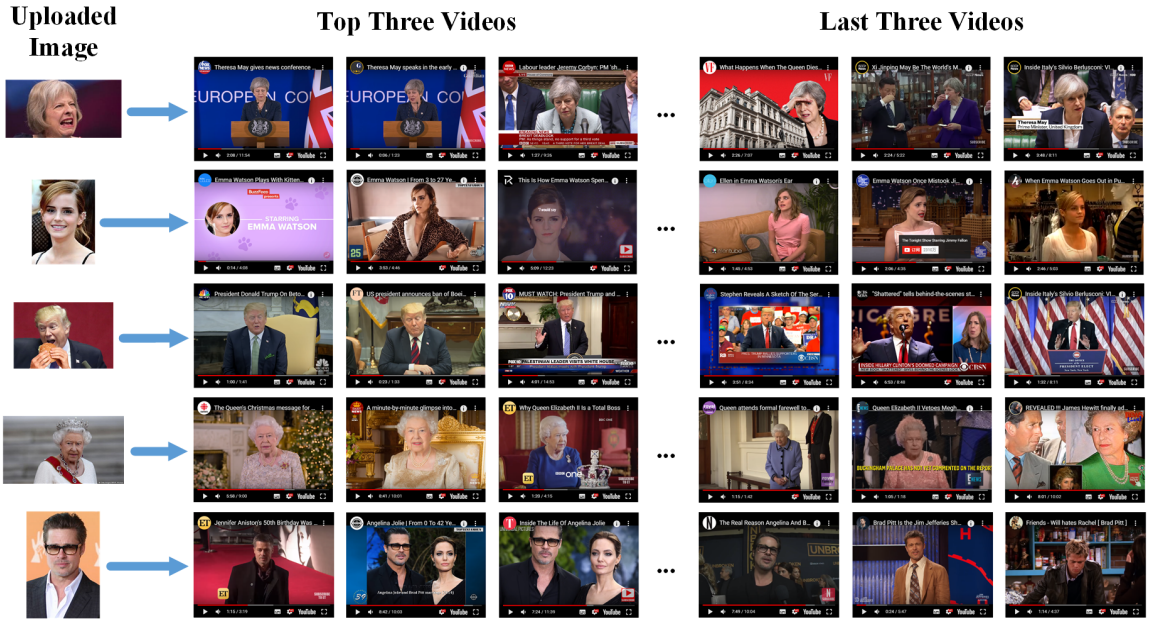


Figure 6.12: Five examples of face search results in videos with the proposed face search framework.

6.6 Summary

This chapter has presented a detailed description of our complete face search framework from its design to its implementation. The proposed face search framework has integrated the state-of-the-art face detection and face recognition methods, and also included all necessary modules. Compared with other frameworks, the proposed framework has three main

features: a) high retrieval accuracy, b) low requirement on uploaded images, and c) support for face search in videos. The evaluation results confirm the above features and show that the proposed framework has very competitive performance on face detection, face recognition and retrieval.

CHAPTER VII

Conclusions and Future Work

This chapter firstly summarises the research of the thesis. The contributions of the methods presented in this thesis are briefly summarised. Then, we introduce our future work and list some relevant and promising research directions.

7.1 Conclusions

Chapter III proposes a multi-descriptor fusion method called MDF, with which we achieved higher face identification accuracy than the benchmark methods. MDF is robust to different types of variance including facial occlusion, illumination variance, expression variance, and pose variance. By combining MDF with KED, we take full advantage of the merits of KED, so MDF+KED can cope with the case of One Sample per Person (OSPP), and just requires a small amount of time to update the whole model when new samples are added into the image gallery. We avoid the demerits, for example, the lack of sufficient discriminant features and robust classifiers. We also propose a general optimisation method called DAMS to search an optimum subset of the feature blocks. DAMS significantly reduces the memory requirement and computational costs of MDF. The new face representation, refined by DAMS, is called Selective Multi-descriptor Fusion (SMDF). Compared with MDF, SMDF has much smaller feature dimension, which results in a much lower configuration requirement. However, SMDF still achieves excellent performance

compared with other methods.

Chapter IV proposes a new supervision signal called Minkowski Distance-based Centre Loss (MC Loss) to improve the Centre Loss. Experimental results on LFW [58] and YTF [178] show that the proposed method achieves higher verification accuracy than the Softmax Loss and the Softmax Loss + Centre Loss, and also achieves competitive results compared with the state-of-the-art methods. Next, Chapter IV proposes Minimum Margin Loss (MML), which aims to force all the class centre pairs to have a distance larger than the specified minimum margin. To the best of our knowledge, MML is the first loss function that considers setting a minimum margin between the class centres. However, it is necessary to have such a constraint to rectify the margin bias introduced by class imbalance in the training data. Experiments are conducted on seven public datasets – LFW [58], SLLFW [34], YTF [178], Megaface [68], FaceScrub [105], IJB-B [174] and IJB-C [100]. Results show that MML achieves better performance than Softmax Loss, Centre Loss, Range Loss and Marginal Loss with almost no increase in computing cost. It also achieved competitive performance compared with the state-of-the-art methods.

Chapter V firstly presents the Precise Adjacent Margin Loss (PAM Loss) to improve the discriminative ability of the deep features. To the best of our knowledge, PAM Loss is the first loss that gives 'margin' a meaning that represents the real margin between the different classes in the training set. Extensive experiments are conducted on public datasets to demonstrate the effectiveness of the proposed PAM Loss. These datasets include LFW [58], YTF [178], MegaFace [68] and FaceScrub [105] datasets. Experimental results verified the state-of-the-art performance of PAM Loss. Then, Chapter V proposes the Global Information-based Cosine Optimal Loss (GICO Loss). It is the first loss function that simultaneously satisfies all the first four properties in Table 5.3 and also the first attempt to use global information as the feedback information in a loss function. To enable the calculation of GICO Loss, we propose an algorithm to learn the cosine similarity between the class centre and the class edge. We conduct extensive experiments on five public

benchmark datasets including LFW [58], SLLFW [195], YTF [178], MegaFace [68] and FaceScrub [105] datasets. Experimental results presented in Section 5.3.5 demonstrate the state-of-the-art performance of GicoFace.

Chapter VI presents a complete face search framework for image and video retrieval. This chapter explains the structure of this framework from its front end to its back end. It also describes the techniques used in each module. This face search framework has integrated the state-of-the-art face detection and face recognition methods, and also included all necessary modules. The proposed framework has three features: a) high retrieval accuracy, b) low requirement on uploaded images and c) support for face search in videos. The experimental results confirm the above features and show that the proposed framework has very competitive performance on face detection, face recognition and retrieval.

7.2 Future Work

The research presented in this thesis focuses only on feature fusion and loss function with application to general face recognition (GFR). However, GFR models may not be able to meet the requirements of some special cases, for example, large pose variation, low resolution, cross-age face recognition and make-up face recognition. Future work will focus on pose-invariant face recognition (PIFR) and low-resolution face recognition (LRFR).

PIFR is the process of recognising a person using face images that are captured under arbitrary poses. PIFR is of significance because it is not always easy for a camera to capture frontal face images in real-world cases, as it is often the case that people show their face under different poses. Many PIFR methods have been proposed in recent years. These methods can be divided into four categories: 1) using pose-invariant features for face representation; 2) projecting features under different poses into a shared subspace; 3) transforming face images from one pose to another pose and matching faces with the same pose; 4) hybrid methods that combine the above three strategies.

A large number of works on general face recognition have already been completed and applied to daily life, for example, the face recognition system developed by Oohashi [112] and the sparse representation based system developed by Andrew *et al.* [158]. However, these methods may be unsatisfactory in some scenarios because they can only cope with near-frontal faces. As in the results listed in [134], many existing face recognition methods suffer a decrease of over 10% when the scene is changed from frontal-frontal verification to frontal-profile verification. Indeed, pose-invariant face recognition is still an unsolved problem, and this provides opportunities for further research [166]. In the future, we will conduct some theoretical research on face pose, trying to find the inherent relation and regularity between face images of the same subject under different poses. Based on the theoretical support, we will then design and train a new network to learn pose-invariant features.

LRFR refers to face recognition using low-resolution facial images. In many real-world applications, the distance between the camera and the subject determines that the resolution of the face images captured by the camera is much lower than for facial images captured in a controlled setting. Even if the images are captured by a 1080P camera, the face portions can be smaller than 25*20 pixels, and such a small image patch contains very little information. Compared with normal face images, low-resolution face portions lose much information that would be useful for classification. Thus, traditional algorithms cannot be used directly to compare low-resolution (LR) images with high-resolution (HR) images, as they do not share a common feature representation.

There are three typical methods for LRFR [84]:

1) Down-sample the HR gallery images. Then perform the matching of LR facial images.

2) Upscale the LR images by super-resolution (SR) or interpolation. Then perform the matching of HR facial images.

3) Project the HR images and LR images into a common subspace. Then perform the matching in the subspace.

Compared with method 1 and method 2, method 3 has advantages, as it neither leads to information loss like method 1 nor brings in external information like method 2. But, how to find an optimum common subspace is a challenging problem for method 3. Different HR/LR image pairs may share quite different subspaces. Therefore, we plan to develop a new network architecture for low-resolution face recognition. To address different cases, this network will be designed to learn a common subspace for all HR/LR image pairs.

REFERENCES

[1] Betaface [Online]. Available: https://www.betaface.com/demo.html.

[2] IntelliVision [Online]. Available: http://www.intelli-vision.com/facial-recognition.

[3] ontotext [Online]. Available: https://www.ontotext.com/knowledgehub/case-studies/smart-surveillance-solution-for-efficient-face-recognition/.

[4] Panasonic [Online]. Available: https://security.panasonic.com/products_technology/technologies/facial_recognition/.

[5] PicTriev [Online]. Available: http://www.pictriev.com/?lang=en.

[6] PimEyes [Online]. Available: http://pimeyes.com/en/.

[7] The results of some baseline methods provided by SLLFW team: http://www.whdeng.cn/sllfw/#results.

[8] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

[9] Timo Ahonen, Abdenour Hadid, and Matti Pietikainen. Face description with local binary patterns: Application to face recognition. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (12):2037–2041, 2006.

[10] Timo Ahonen, Esa Rahtu, Ville Ojansivu, and Janne Heikkila. Recognition of blurred faces using local phase quantization. In *2008 19th international conference on pattern recognition*, pages 1–4. IEEE, 2008.

[11] Alexandr Andoni and Piotr Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Communications of the ACM*, 51(1):117, 2008.

[12] Bowen Baker, Otkrist Gupta, Nikhil Naik, and Ramesh Raskar. Designing neural network architectures using reinforcement learning. In *5th International Conference on Learning Representations (ICLR)*, 2017.

[13] Marian Stewart Bartlett. Independent component representations for face recognition. In *Face Image Analysis by Unsupervised Learning*, pages 39–67. Springer, 2001.

[14] Marian Stewart Bartlett, Gwen Littlewort, Mark G Frank, Claudia Lainscsek, Ian R Fasel, Javier R Movellan, et al. Automatic recognition of facial actions in spontaneous expressions. *Journal of multimedia*, 1(6):22–35, 2006.

[15] Gaston Baudat and Fatiha Anouar. Generalized discriminant analysis using a kernel approach. *Neural computation*, 12(10):2385–2404, 2000.

[16] Peter N Belhumeur, João P Hespanha, and David J Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (7):711–720, 1997.

[17] Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In *Advances in neural information processing systems*, pages 153–160, 2007.

[18] Lacey Best-Rowden, Hu Han, Charles Otto, Brendan F Klare, and Anil K Jain. Unconstrained face recognition: Identifying a person of interest from a media collection. *IEEE Transactions on Information Forensics and Security*, 9(12):2144–2157, 2014.

[19] Djamel Bouchaffra and Abbes Amira. Structural hidden markov models for biometrics: Fusion of face and fingerprint. *Pattern Recognition*, 41(3):852–867, 2008.

[20] Emmanuel J Candès, Justin Romberg, and Terence Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on information theory*, 52(2):489–509, 2006.

[21] Qiong Cao, Li Shen, Weidi Xie, Omkar M Parkhi, and Andrew Zisserman. Vggface2: A dataset for recognising faces across pose and age. *arXiv preprint arXiv:1710.08092*, 2017.

[22] Chi Ho Chan, Muhammad Atif Tahir, Josef Kittler, and Matti Pietikäinen. Multiscale local phase quantization for robust component-based face recognition using kernel fusion of multiple descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(5):1164–1177, 2012.

[23] Binghui Chen, Weihong Deng, and Junping Du. Noisy softmax: Improving the generalization ability of dcnn via postponing the early softmax saturation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[24] Lifen Chen, Hongyuan Mark Liao, Mingtat Ko, Jachen Lin, and Gwojong Yu. A new lda-based face recognition system which can solve the small sample size problem. *Pattern recognition*, 33(10):1713–1726, 2000.

[25] Line Clemmensen, Trevor Hastie, Daniela Witten, and Bjarne Ersbøll. Sparse discriminant analysis. *Technometrics*, 53(4):406–413, 2011.

[26] Timothy F Cootes, Gareth J Edwards, and Christopher J Taylor. Active appearance models. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (6):681–685, 2001.

[27] Timothy F Cootes, Christopher J Taylor, David H Cooper, and Jim Graham. Active shape models-their training and application. *Computer vision and image understanding*, 61(1):38–59, 1995.

[28] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.

[29] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2019.

[30] Jiankang Deng, Yuxiang Zhou, and Stefanos Zafeiriou. Marginal loss for deep face recognition. pages 60–68, 2017.

[31] Weihong Deng, Jiani Hu, and Jun Guo. Extended src: Undersampled face recognition via intraclass variant dictionary. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(9):1864–1870, 2012.

[32] Weihong Deng, Jiani Hu, and Jun Guo. In defense of sparsity based face recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013.

[33] Weihong Deng, Jiani Hu, Jiwen Lu, and Jun Guo. Transform-invariant pca: A unified approach to fully automatic facealignment, representation, and recognition. *IEEE transactions on pattern analysis and machine intelligence*, 36(6):1275–1284, 2013.

[34] Weihong Deng, Jiani Hu, Nanhai Zhang, Binghui Chen, and Jun Guo. Fine-grained face verification: Fglfw database, baselines, and human-dcmn partnership. *Pattern Recognition*, 66:63–73, 2017.

[35] Changxing Ding, Jonghyun Choi, Dacheng Tao, and Larry S Davis. Multi-directional multi-level dual-cross patterns for robust face recognition. *IEEE transactions on pattern analysis and machine intelligence*, 38(3):518–531, 2015.

[36] Changxing Ding and Dacheng Tao. Robust face recognition via multimodal deep face representation. *IEEE Transactions on Multimedia*, 17(11):2049–2058, 2015.

[37] Changxing Ding and Dacheng Tao. Trunk-branch ensemble convolutional neural networks for video-based face recognition. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):1002–1014, 2017.

[38] Meng Joo Er, Shiqian Wu, Juwei Lu, and Hock Lye Toh. Face recognition with radial basis function (rbf) neural networks. *IEEE transactions on neural networks*, 13(3):697–710, 2002.

[39] Lunke Fei, Guangming Lu, Wei Jia, Shaohua Teng, and David Zhang. Feature extraction methods for palmprint recognition: A survey and evaluation. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 49(2):346–363, 2018.

[40] Lunke Fei, Yong Xu, Wenliang Tang, and David Zhang. Double-orientation code and nonlinear matching scheme for palmprint recognition. *Pattern Recognition*, 49:89–101, 2016.

[41] Atoany N Fierro-Radilla, Mariko Nakano-Miyatake, Hector Perez-Meana, Manuel Cedillo-Hernandez, and Francisco Garcia-Ugalde. An efficient color descriptor based on global and local color features for image retrieval. In *Electrical Engineering, Computing Science and Automatic Control (CCE), 2013 10th International Conference on*, pages 233–238. IEEE, 2013.

[42] Dario Floreano, Peter Dürr, and Claudio Mattiussi. Neuroevolution: from architectures to learning. *Evolutionary intelligence*, 1(1):47–62, 2008.

[43] Wen Gao, Bo Cao, Shiguang Shan, Xilin Chen, Delong Zhou, Xiaohua Zhang, and Debin Zhao. The cas-peal large-scale chinese face database and baseline evaluations. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 38(1):149–161, 2007.

[44] Zhirong Gao, Lixin Ding, Chengyi Xiong, and Bo Huang. A robust face recognition method using multiple features fusion and linear regression. *Wuhan University Journal of Natural Sciences*, 19(4):323–327, 2014-08-01.

[45] Athinodoros S Georghiades, Peter N Belhumeur, and David J Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (6):643–660, 2001.

[46] Yunchao Gong, Svetlana Lazebnik, Albert Gordo, and Florent Perronnin. Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(12):2916–2929, 2012.

[47] Ralph Gross, Iain Matthews, Jeffrey Cohn, Takeo Kanade, and Simon Baker. Multi-pie. *Image and Vision Computing*, 28(5):807–813, 2010.

[48] Yandong Guo, Lei Zhang, Yuxiao Hu, Xiaodong He, and Jianfeng Gao. MS-Celeb-1M: A dataset and benchmark for large scale face recognition. In *European Conference on Computer Vision*. Springer, 2016.

[49] Chunrui Han, Shiguang Shan, Meina Kan, Shuzhe Wu, and Xilin Chen. Face recognition with contrastive convolution. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[50] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[51] Bernd Heisele, Purdy Ho, and Tomaso Poggio. Face recognition with support vector machines: Global versus component-based approach. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 2, pages 688–694. IEEE, 2001.

[52] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[53] Erik Hjelmås and Boon Kee Low. Face detection: A survey. *Computer vision and image understanding*, 83(3):236–274, 2001.

[54] Rein-Lien Hsu, Mohamed Abdel-Mottaleb, and Anil K Jain. Face detection in color images. *IEEE transactions on pattern analysis and machine intelligence*, 24(5):696–706, 2002.

[55] Guosheng Hu, Yongxin Yang, Dong Yi, Josef Kittler, William Christmas, Stan Z Li, and Timothy Hospedales. When face recognition meets with deep learning: an evaluation of convolutional neural networks for face recognition. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 142–150, 2015.

[56] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[57] Gary B Huang and Erik Learned-Miller. Labeled faces in the wild: Updates and new reporting procedures. *Dept. Comput. Sci., Univ. Massachusetts Amherst, Amherst, MA, USA, Tech. Rep*, pages 14–003, 2014.

[58] Gary B Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.

[59] Kekun Huang, Daoqing Dai, Chuanxian Ren, and Zhaorong Lai. Learning kernel extended dictionary for face recognition. *IEEE transactions on neural networks and learning systems*, 28(5):1082–1094, 2016.

[60] Kekun Huang, Daoqing Dai, Chuanxian Ren, Yufeng Yu, and Zhaorong Lai. Fusing landmark-based features at kernel level for face recognition. *Pattern Recognition*, 63:406–415, 2017-03.

[61] David H Hubel and Torsten N Wiesel. Receptive fields of single neurones in the cat's striate cortex. *The Journal of physiology*, 148(3):574–591, 1959.

[62] Rabia Jafri and Hamid R Arabnia. A survey of face recognition techniques. *Journal of Information Processing Systems*, 5(2):41–68, 2009.

[63] Anil K Jain, Arun Ross, and Salil Prabhakar. An introduction to biometric recognition. *IEEE Transactions on circuits and systems for video technology*, 14(1), 2004.

[64] Ian T Jolliffe and Jorge Cadima. Principal component analysis: a review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202, 2016.

[65] Meina Kan, Shiguang Shan, and Xilin Chen. Multi-view deep network for cross-view classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4847–4855, 2016.

[66] Paweł Karczmarek, Adam Kiersztyn, Witold Pedrycz, and Michał Dolecki. An application of chain code-based local descriptor and its extension to face recognition. *Pattern Recognition*, 65:26–34, 2017.

[67] Juha Karhunen. Principal component neural networks—theory and applications. *Pattern Analysis & Applications*, 1(1):74–75, 1998.

[68] Ira Kemelmacher-Shlizerman, Steven M Seitz, Daniel Miller, and Evan Brossard. The megaface benchmark: 1 million faces for recognition at scale. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4873–4882, 2016.

[69] Joongrock Kim, Sunjin Yu, Dongchul Kim, Kar-Ann Toh, and Sangyoun Lee. An adaptive local binary pattern for 3d hand tracking. *Pattern Recognition*, 61(Supplement C):139–152, January 2017.

[70] Kwang I Kim, Keechul Jung, and Hang J Kim. Face recognition using kernel principal component analysis. *IEEE signal processing letters*, 9(2):40–42, 2002.

[71] Piotr Koniusz, Fei Yan, Philippe-Henri Gosselin, and Krystian Mikolajczyk. Higher-order occurrence pooling for bags-of-words: Visual concept detection. *IEEE transactions on pattern analysis and machine intelligence*, 39(2):313–326, 2016.

[72] Piotr Koniusz, Fei Yan, and Krystian Mikolajczyk. Comparison of mid-level feature coding approaches and pooling strategies in visual concept detection. *Computer vision and image understanding*, 117(5):479–492, 2013.

[73] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[74] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[75] Neeraj Kumar, Alexander Berg, Peter N Belhumeur, and Shree Nayar. Describable visual attributes for face verification and image search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(10):1962–1977, 2011.

[76] Neeraj Kumar, Alexander C Berg, Peter N Belhumeur, and Shree K Nayar. Attribute and simile classifiers for face verification. In *2009 IEEE 12th international conference on computer vision*, pages 365–372. IEEE, 2009.

[77] Martin Lades, Jan C Vorbruggen, Joachim Buhmann, Jörg Lange, Christoph Von Der Malsburg, Rolf P Wurtz, and Wolfgang Konen. Distortion invariant object recognition in the dynamic link architecture. *IEEE Transactions on computers*, (3):300–311, 1993.

[78] Rushi Lan, Yicong Zhou, and Yuan Yan Tang. Quaternionic local ranking binary pattern: a local descriptor of color images. *IEEE Transactions on Image Processing*, 25(2):566–579, 2016.

[79] Steve Lawrence, C Lee Giles, Ah Chung Tsoi, and Andrew D Back. Face recognition: A convolutional neural-network approach. *IEEE transactions on neural networks*, 8(1):98–113, 1997.

[80] Erik Learned-Miller, Gary B. Huang, Aruni RoyChowdhury, Haoxiang Li, and Gang Hua. Labeled faces in the wild: A survey. In Michal Kawulok, M. Emre Celebi, and Bogdan Smolka, editors, *Advances in Face Detection and Facial Image Analysis*, pages 189–248. Springer International Publishing, 2016. DOI: 10.1007/978-3-319-25958-1_8.

[81] Yann Lecun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[82] Wei Li, Chen Chen, Hongjun Su, and Qian Du. Local binary patterns and extreme learning machine for hyperspectral imagery classification. *IEEE Transactions on Geoscience and Remote Sensing*, 53(7):3681–3693, 2015.

[83] Xiang Li, Wenhai Wang, Xiaolin Hu, and Jian Yang. Selective kernel networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 510–519, 2019.

[84] Xiaoxin Li, Daoqing Dai, Xiaofei Zhang, and Chuanxian Ren. Structured sparse error coding for face recognition with occlusion. *IEEE transactions on image processing*, 22(5):1889–1900, 2013.

[85] Jingtuo Liu, Yafeng Deng, Tao Bai, Zhengping Wei, and Chang Huang. Targeting ultimate accuracy: Face recognition via deep embedding. *arXiv preprint arXiv:1506.07310*, 2015.

[86] Jingtuo Liu, Yafeng Deng, Tao Bai, Zhengping Wei, and Chang Huang. Targeting ultimate accuracy: Face recognition via deep embedding. *arXiv:1506.07310 [cs]*, June 2015. arXiv: 1506.07310.

[87] Li Liu, Shu Wang, Guoxin Su, Zi-Gang Huang, and Ming Liu. Towards complex activity recognition using a bayesian network-based probabilistic generative framework. *Pattern Recognition*, 68:295–309, 2017.

[88] Qingshan Liu, Hanqing Lu, and Songde Ma. Improving kernel fisher discriminant analysis for face recognition. *IEEE transactions on circuits and systems for video technology*, 14(1):42–49, 2004.

[89] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6738–6746, Honolulu, HI, July 2017. IEEE.

[90] Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang. Large-margin softmax loss for convolutional neural networks. In *International Conference on Machine Learning*, pages 507–516, June 2016.

[91] Weiyang Liu, Yan-Ming Zhang, Xingguo Li, Zhiding Yu, Bo Dai, Tuo Zhao, and Le Song. Deep hyperspherical learning. In *Advances in Neural Information Processing Systems*, pages 3950–3960, 2017.

[92] Yu Liu, Hongyang Li, and Xiaogang Wang. Rethinking feature discrimination and polymerization for large-scale recognition. *arXiv preprint arXiv:1710.00870*, 2017.

[93] Cheng-Yaw Low, Andrew Beng-Jin Teoh, and Cong-Jie Ng. Multi-fold gabor filter convolution descriptor for face recognition. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2094–2098. IEEE, 2016.

[94] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[95] Guifu Lu, Jian Zou, and Yong Wang. Incremental complete lda for face recognition. *Pattern Recognition*, 45(7):2510–2521, 2012.

[96] Jianming Lu, Xue Yuan, and Takashi Yahagi. A method of face recognition based on fuzzy clustering and parallel neural networks. *Signal Processing*, 86(8):2026–2039, 2006.

[97] Utthara Gosa Mangai, Suranjana Samanta, Sukhendu Das, and Pinaki Roy Chowdhury. A survey of decision fusion and feature fusion strategies for pattern classification. *IETE Technical Review*, 27(4):293–307, July 2010.

[98] Iacopo Masi, Stephen Rawls, Gérard Medioni, and Prem Natarajan. Pose-aware face recognition in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4838–4846, 2016.

[99] Iacopo Masi, Anh Tuan Tran, Tal Hassner, Jatuporn Toy Leksut, and Gérard Medioni. Do we really need to collect millions of faces for effective face recognition? In *Computer Vision – ECCV 2016*, Lecture Notes in Computer Science, pages 579–596. Springer, Cham, October 2016.

[100] Brianna Maze, Jocelyn Adams, James A Duncan, Nathan Kalka, Tim Miller, Charles Otto, Anil K Jain, W Tyler Niggel, Janet Anderson, Jordan Cheney, et al. Iarpa janus benchmark-c: Face dataset and protocol. In *2018 International Conference on Biometrics (ICB)*, pages 158–165. IEEE, 2018.

[101] Patricia Melin, Cristina Felix, and Oscar Castillo. Face recognition using modular neural networks and the fuzzy sugeno integral for response integration. *International Journal Of Intelligent Systems*, 20(2):275–291, 2005.

[102] Hwang-Ki Min, Yuxi Hou, Sangwoo Park, and Iickho Song. A computationally efficient scheme for feature extraction with kernel discriminant analysis. *Pattern Recognition*, 50:45–55, 2016.

[103] Hae-Min Moon, Chang Ho Seo, and Sung Bum Pan. A face recognition system based on convolution neural network using multiple distance face. *Soft Computing*, 21(17):4995–5002, 2017.

[104] Ara V Nefian and Monson H Hayes. Hidden markov models for face recognition. In *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'98 (Cat. No. 98CH36181)*, volume 5, pages 2721–2724. IEEE, 1998.

[105] Hongwei Ng and Stefan Winkler. A data-driven approach to cleaning large face datasets. In *Image Processing (ICIP), 2014 IEEE International Conference on*, pages 343–347. IEEE, 2014.

[106] Soodeh Nikan and Majid Ahmadi. Local gradient-based illumination invariant face recognition using local phase quantisation and multi-resolution local binary pattern fusion. *IET Image Processing*, 9(1):12–21, 2014-07-16.

[107] Beom-Seok Oh, Kangrok Oh, Andrew Beng Jin Teoh, Zhiping Lin, and Kar-Ann Toh. A gabor-based network for heterogeneous face recognition. *Neurocomputing*, 261:253–265, 2017.

[108] Timo Ojala, Matti Pietikäinen, and David Harwood. A comparative study of texture measures with classification based on featured distributions. *Pattern recognition*, 29(1):51–59, 1996.

[109] Timo Ojala, Matti Pietikainen, and Topi Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(7):971–987, 2002.

[110] Timo Ojala, Matti Pietikainen, and Topi Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on pattern analysis and machine intelligence*, 24(7):971–987, 2002.

[111] Bruno A Olshausen and David J Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607, 1996.

[112] Takahiro Oohashi. Face identification system, 2006. US Patent 7,014,102.

[113] Yanwei Pang, Yuan Yuan, Xuelong Li, and Jing Pan. Efficient hog human detection. *Signal Processing*, 91(4):773–781, 2011.

[114] Omkar M. Parkhi, Andrea Vedaldi, and Andrew Zisserman. Deep face recognition. In *2015 British Machine Vision Conference (BMVC)*, volume 1, page 6, September 2015.

[115] Xi Peng, Xiang Yu, Kihyuk Sohn, Dimitris N Metaxas, and Manmohan Chandraker. Reconstruction-based disentanglement for pose-invariant face recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 1623–1632, 2017.

[116] Jonathon Phillips, Patrick Grother, Ross Micheals, Duane M Blackburn, Elham Tabassi, and Mike Bone. Face recognition vendor test 2002. In *2003 IEEE International SOI Conference. Proceedings (Cat. No. 03CH37443)*, page 44. IEEE, 2003.

[117] Bruce Poon, Ashraful Amin, and Hong Yan. Gabor phase representation on human face recognition for distorted images. In *2016 International Conference on Wavelet Analysis and Pattern Recognition (ICWAPR)*, pages 276–281. IEEE, 2016.

[118] Xianbiao Qi, Rong Xiao, Chun-Guang Li, Yu Qiao, Jun Guo, and Xiaoou Tang. Pairwise rotation invariant co-occurrence local binary pattern. *IEEE transactions on pattern analysis and machine intelligence*, 36(11):2199–2213, 2014.

[119] Rajeev Ranjan, Ankan Bansal, Jingxiao Zheng, Hongyu Xu, Joshua Gleason, Boyu Lu, Anirudh Nanduri, Juncheng Chen, Carlos D Castillo, and Rama Chellappa. A fast and accurate system for face detection, identification, and verification. *IEEE Transactions on Biometrics, Behavior, and Identity Science*, 1(2):82–96, 2019.

[120] Rajeev Ranjan, Carlos D. Castillo, and Rama Chellappa. L2-constrained softmax loss for discriminative face verification. March 2017.

[121] Rajeev Ranjan, Swami Sankaranarayanan, Carlos D Castillo, and Rama Chellappa. An all-in-one convolutional neural network for face analysis. In *2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*, pages 17–24. IEEE, 2017.

[122] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. Regularized evolution for image classifier architecture search. In *Proceedings of the aaai conference on artificial intelligence*, volume 33, pages 4780–4789, 2019.

[123] Esteban Real, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Jie Tan, Quoc V Le, and Alexey Kurakin. Large-scale evolution of image classifiers. In *34th International Conference on Machine Learning*, pages 2902–2911, 2017.

[124] Chuanxian Ren, Daoqing Dai, Xiaoxin Li, and Zhaorong Lai. Band-reweighed gabor kernel embedding for face image representation and recognition. *IEEE Transactions on Image Processing*, 23(2):725–740, 2013.

[125] Seon-Min Rhee, Byung-In Yoo, Jae-Joon Han, and Wonjun Hwang. Deep neural network using color and synthesized three-dimensional shape for face recognition. *Journal of Electronic Imaging*, 26(2):020502, 2017.

[126] Henry A Rowley, Shumeet Baluja, and Takeo Kanade. Neural network-based face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (1):23–38, 1998.

[127] Salvatore Ruggieri. Efficient C4.5 [classification algorithm]. *IEEE Transactions on Knowledge and Data Engineering*, 14(2):438–444, March 2002.

[128] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1988.

[129] Ferdinando Samaria and Steve Young. Hmm-based architecture for face identification. *Image and vision computing*, 12(8):537–543, 1994.

[130] Jason Saragih. Principal regression analysis. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 2881–2888. IEEE, 2011.

[131] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[132] Florian Schroff, Dmitry Kalenichenko, and James Philbin. FaceNet: A unified embedding for face recognition and clustering. pages 815–823, 2015.

[133] Nicu Sebe, Michael S Lew, Ira Cohen, Ashutosh Garg, and Thomas S Huang. Emotion recognition using a cauchy naive bayes classifier. In *Object recognition supported by user interaction for service robots*, volume 1, pages 17–20. IEEE, 2002.

[134] Soumyadip Sengupta, Juncheng Chen, Carlos Castillo, Vishal M Patel, Rama Chellappa, and David W Jacobs. Frontal to profile face verification in the wild. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–9. IEEE, 2016.

[135] Maliheh Shabanzade, Morteza Zahedi, and Seyyed A Aghvami. Combination of local descriptors and global features for leaf recognition. *Signal & Image Processing*, 2(3):23, 2011.

[136] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[137] Guangda Su, Cuiping Zhang, Rong Ding, and Cheng Du. Mmp-pca face recognition method. *Electronics Letters*, 38(25):1654–1656, 2002.

[138] Pratibha Sukhija, Sunny Behal, and Pritpal Singh. Face recognition system using genetic algorithm. *Procedia Computer Science*, 85:410–417, 2016.

[139] Yi Sun, Yuheng Chen, Xiaogang Wang, and Xiaoou Tang. Deep learning face representation by joint identification-verification. In *Advances in neural information processing systems*, pages 1988–1996, 2014.

[140] Yi Sun, Ding Liang, Xiaogang Wang, and Xiaoou Tang. Deepid3: Face recognition with very deep neural networks. *arXiv preprint arXiv:1502.00873*, 2015.

[141] Yi Sun, Xiaogang Wang, and Xiaoou Tang. Deep learning face representation from predicting 10,000 classes. pages 1891–1898, 2014.

[142] Yi Sun, Xiaogang Wang, and Xiaoou Tang. Deeply learned face representations are sparse, selective, and robust. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2892–2900, 2015.

[143] Daniel L Swets and John Juyang Weng. Using discriminant eigenfeatures for image retrieval. *IEEE Transactions on pattern analysis and machine intelligence*, 18(8):831–836, 1996.

[144] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-first AAAI conference on artificial intelligence*, 2017.

[145] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[146] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[147] Oren Tadmor, Tal Rosenwein, Shai Shalev-Shwartz, Yonatan Wexler, and Amnon Shashua. Learning a metric embedding for face recognition using the multibatch method. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, pages 1396–1397, USA, 2016. Curran Associates Inc.

[148] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '14, pages 1701–1708, Washington, DC, USA, 2014. IEEE Computer Society.

[149] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Web-scale training for face identification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2746–2754, 2015.

[150] Hengliang Tan, Bing Yang, and Zhengming Ma. Face recognition based on the fusion of global and local hog features of face images. *IET computer vision*, 8(3):224–234, 2013.

[151] Xiaoyang Tan and Bill Triggs. Enhanced local texture feature sets for face recognition under difficult lighting conditions. *IEEE transactions on image processing*, 19(6):1635–1650, 2010.

[152] Xiaoyang Tan and William Triggs. Enhanced local texture feature sets for face recognition under difficult lighting conditions. *IEEE transactions on image processing*, 19(6):1635–1650, 2010.

[153] Jirí Trefnỳ and Jirí Matas. Extended set of local binary patterns for rapid object detection. In *Computer Vision Winter Workshop*, pages 1–7, 2010.

[154] Matthew Turk and Alex Pentland. Eigenfaces for recognition. *Journal of cognitive neuroscience*, 3(1):71–86, 1991.

[155] Tõnis Uiboupin, Pejman Rasti, Gholamreza Anbarjafari, and Hasan Demirel. Facial image super resolution using sparse representation for improving face recognition in surveillance monitoring. In *2016 24th Signal Processing and Communication Application Conference (SIU)*, pages 437–440. IEEE, 2016.

[156] Paul Viola and Michael J Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004.

[157] Ngoc-Son Vu and Alice Caplier. Enhanced patterns of oriented edge magnitudes for face recognition and image matching. *IEEE Transactions on Image Processing*, 21(3):1352–1365, 2011.

[158] Andrew Wagner, John Wright, Arvind Ganesh, Zihan Zhou, Hossein Mobahi, and Yi Ma. Toward a practical face recognition system: Robust alignment and illumination by sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(2):372–386, 2011.

[159] Dayong Wang, Charles Otto, and Anil K Jain. Face search at scale. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1122–1136, 2016.

[160] Feng Wang, Jian Cheng, Weiyang Liu, and Haijun Liu. Additive margin softmax for face verification. *IEEE Signal Processing Letters*, 25(7):926–930, 2018.

[161] Feng Wang, Xiang Xiang, Jian Cheng, and Alan Loddon Yuille. Normface: L_2 hypersphere embedding for face verification. In *Proceedings of the 2017 ACM on Multimedia Conference - MM '17*, pages 1041–1049, Mountain View, California, USA, 2017. ACM Press.

[162] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. *arXiv:1801.09414 [cs]*, January 2018. arXiv: 1801.09414.

[163] Jiakailin Wang, Jinjin Zheng, Shiwu Zhang, Jijun He, Xiao Liang, and Sui Feng. A face recognition system based on local binary patterns and support vector machine for home security service robot. In *2016 9th International Symposium on Computational Intelligence and Design (ISCID)*, volume 2, pages 303–307. IEEE, 2016.

[164] Jun Wang, Sanjiv Kumar, and Shih-Fu Chang. Semi-supervised hashing for scalable image retrieval. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.

[165] Linnan Wang, Yiyang Zhao, Yuu Jinnai, Yuandong Tian, and Rodrigo Fonseca. Alphax: exploring neural architectures with deep neural networks and monte carlo tree search. *arXiv preprint arXiv:1903.11059*, 2019.

[166] Mei Wang and Weihong Deng. Deep face recognition: A survey. *arXiv preprint arXiv:1804.06655*, 2018.

[167] Wen Wang, Zhen Cui, Hong Chang, Shiguang Shan, and Xilin Chen. Deeply coupled auto-encoder networks for cross-view classification. *arXiv preprint arXiv:1402.2031*, 2014.

[168] Yitong Wang, Dihong Gong, Zheng Zhou, Xing Ji, Hao Wang, Zhifeng Li, Wei Liu, and Tong Zhang. Orthogonal deep features decomposition for age-invariant face recognition. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[169] Tomoki Watanabe, Satoshi Ito, and Kentaro Yokoi. Co-occurrence histograms of oriented gradients for human detection. *IPSJ Transactions on Computer Vision and Applications*, 2:39–47, 2010.

[170] Xin Wei, Hui Wang, Gongde Guo, and Huan Wan. Multiplex image representation for enhanced recognition. *International Journal of Machine Learning and Cybernetics*, pages 1–10, 2015-09-21.

[171] Kilian Q Weinberger and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10(Feb):207–244, 2009.

[172] Yair Weiss, Antonio Torralba, and Rob Fergus. Spectral hashing. In *Advances in neural information processing systems*, pages 1753–1760, 2009.

[173] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *Computer Vision – ECCV 2016*, Lecture Notes in Computer Science, pages 499–515. Springer, Cham, October 2016.

[174] Cameron Whitelam, Emma Taborsky, Austin Blanton, Brianna Maze, Jocelyn Adams, Tim Miller, Nathan Kalka, Anil K Jain, James A Duncan, Kristen Allen, Jordan Cheney, and Patrick Grother. Iarpa janus benchmark-b face dataset. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017.

[175] Wikipedia contributors. Visual descriptor — Wikipedia, the free encyclopedia, 2018. [Online; accessed 9-September-2019].

[176] Wikipedia contributors. Deep learning — Wikipedia, the free encyclopedia, 2019. [Online; accessed 9-September-2019].

[177] Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52, 1987.

[178] Lior Wolf, Tal Hassner, and Itay Maoz. Face recognition in unconstrained videos with matched background similarity. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 529–534. IEEE, 2011.

[179] John Wright, Allen Y Yang, Arvind Ganesh, S Shankar Sastry, and Yi Ma. Robust face recognition via sparse representation. *IEEE transactions on pattern analysis and machine intelligence*, 31(2):210–227, 2008.

[180] Shufu Xie, Shiguang Shan, Xilin Chen, and Jie Chen. Fusing local patterns of gabor magnitude and phase for face recognition. *IEEE transactions on image processing*, 19(5):1349–1361, 2010.

[181] Weidi Xie, Li Shen, and Andrew Zisserman. Comparator networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 782–797, 2018.

[182] Weidi Xie and Andrew Zisserman. Multicolumn networks for face recognition. *arXiv preprint:1807.09192v1, 24 Jul 2018*, 2018.

[183] Xuehan Xiong and Fernando De la Torre. Supervised descent method and its applications to face alignment. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 532–539, 2013.

[184] Niannan Xue, Jiankang Deng, Shiyang Cheng, Yannis Panagakis, and Stefanos Zafeiriou. Side information for face completion: a robust pca approach. *IEEE transactions on pattern analysis and machine intelligence*, 41(10):2349–2364, 2019.

[185] Pu Yan, Dong Liang, Jun Tang, and Ming Zhu. Local feature descriptor using entropy rate. *Neurocomputing*, 194:157–167, 2016.

[186] Bin Yang, Junjie Yan, Zhen Lei, and Stan Z Li. Aggregate channel features for multi-view face detection. In *IEEE international joint conference on biometrics*, pages 1–8. IEEE, 2014.

[187] Meng Yang, Lei Zhang, Simon Chi-Keung Shiu, and David Zhang. Robust kernel representation with statistical local features for face recognition. *IEEE transactions on neural networks and learning systems*, 24(6):900–912, 2013.

[188] Meng Yang, Lei Zhang, Jian Yang, and David Zhang. Regularized robust coding for face recognition. *IEEE transactions on image processing*, 22(5):1753–1766, 2012.

[189] Shuo Yang, Ping Luo, Chen-Change Loy, and Xiaoou Tang. Wider face: A face detection benchmark. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5525–5533, 2016.

[190] Jieping Ye, Ravi Janardan, and Qi Li. Two-dimensional linear discriminant analysis. In *Advances in neural information processing systems*, pages 1569–1576, 2005.

[191] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z Li. Learning face representation from scratch. *arXiv preprint arXiv:1411.7923*, 2014.

[192] Juha Ylioinas, Norman Poh, Jukka Holappa, and Matti Pietikäinen. Data-driven techniques for smoothing histograms of local binary patterns. *Pattern Recognition*, 60:734–747, 2016.

[193] Dong-jun Yu, Hai-tao Zhao, and Jing-yu Yang. Face recognition: An approach based on feature fusion and neural network [j]. *Acta Simulata Systematica Sinica*, 5, 2005.

[194] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, 23(10):1499–1503, 2016.

[195] Nanhai Zhang and Weihong Deng. Fine-grained lfw database. In *International Conference on Biometrics*, pages 1–6, 2016.

[196] X. Zhang, Z. Fang, Y. Wen, Z. Li, and Y. Qiao. Range loss for deep face recognition with long-tailed training data. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 5419–5428, October 2017.

[197] Zhongqiu Zhao, Deshuang Huang, and Bingyu Sun. Human face recognition based on multi-features using neural networks committee. *Pattern Recognition Letters*, 25(12):1351–1358, 2004.

[198] Xiantong Zhen, Feng Zheng, Ling Shao, Xianbin Cao, and Dan Xu. Supervised local descriptor learning for human action recognition. *IEEE Transactions on Multimedia*, 19(9):2056–2065, 2017.

[199] Erjin Zhou, Zhimin Cao, and Qi Yin. Naive-deep face recognition: Touching the limit of lfw benchmark or not? *arXiv preprint arXiv:1501.04690*, 2015.

[200] Lawrence Zitnick and Piotr Dollár. Edge boxes: Locating object proposals from edges. In *European conference on computer vision*, pages 391–405. Springer, 2014.

[201] Barret Zoph and Quoc Le. Neural architecture search with reinforcement learning. In *5th International Conference on Learning Representations (ICLR)*, 2017.